ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Laboratorio di

Sicurezza Informatica

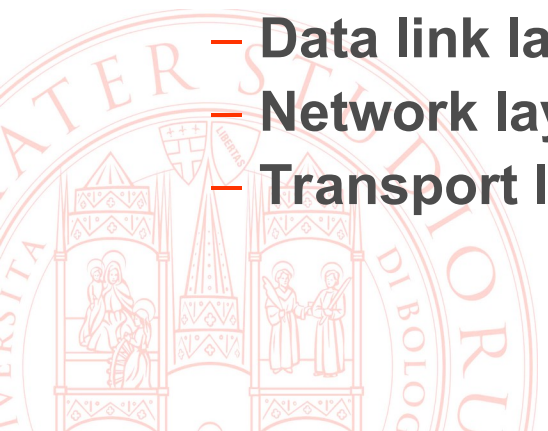
Protezione delle comunicazioni

Marco Prandini

Dipartimento di Informatica – Scienza e Ingegneria

Outline

- Richiami di reti
- Attacchi Passivi
  - Scanning
  - Sniffing
  - Wireless key recovery
- Attacchi Attivi
  - Hijacking ai diversi layer: ARP, BGP, DNS, HTTP
    - e un esempio di come procedere nella kill chain: HTTPS splitting and stripping
  - (D)DoS
- Contromisure: canali sicuri
  - Data link layer: VLANs
  - Network layer: IPSec
  - Transport layer: TLS

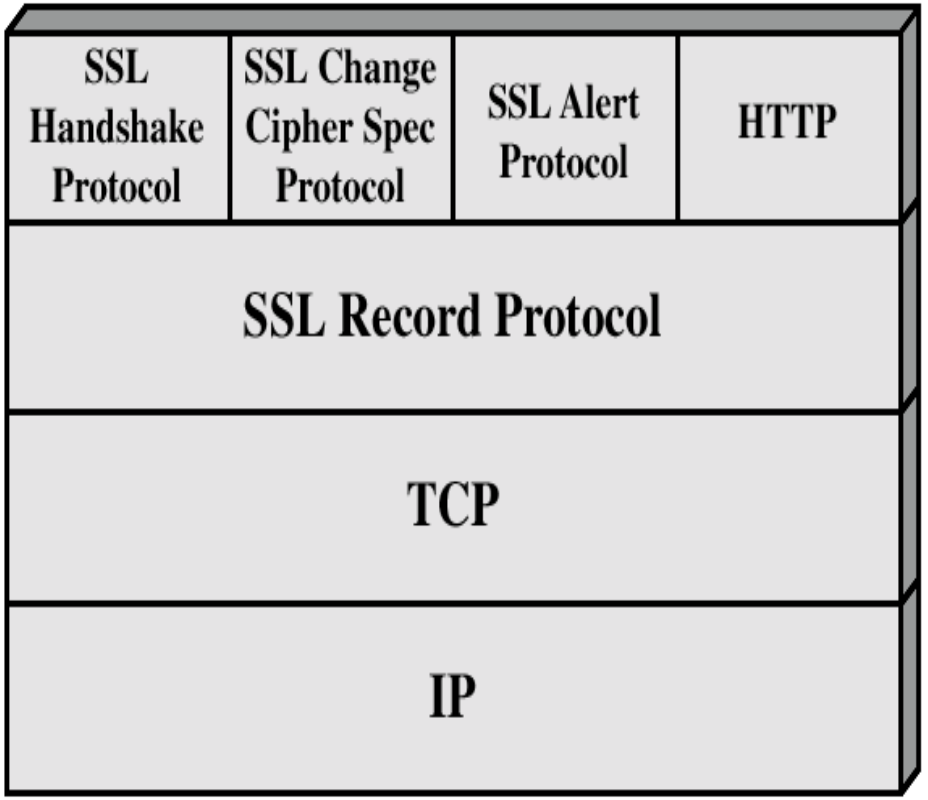


Contromisure e contro-contromisure

- Protezione a livello applicativo: il caso di HTTPS
  - HTTP over SSL (ora TLS)
  - permette di bloccare questi attacchi ma esistono modi
    - per evitare che venga visualizzata l'URL effettivamente visitata
    - o per far accettare al browser qualsiasi certificato
- Protezione a livello di rete: IPSec
- Protezione del data link layer
- Strumenti di uso comune: TOR e SSH

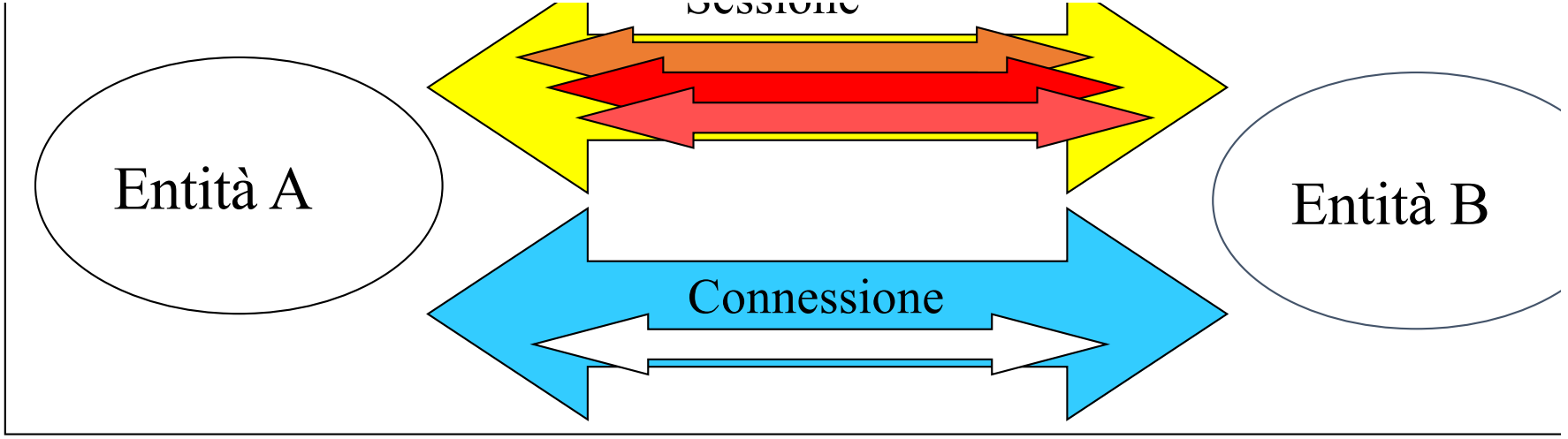
SSL/TLS

- SSL è stato progettato come uno strato di protocolli indipendente
- Si colloca logicamente fra strato di trasporto e applicazioni
  - Grazie a questa architettura non richiede una modifica dei protocolli di rete
- L'implementazione si presenta come una serie di funzioni di libreria (SSLeay, OpenSSL, ...)
  - Per rendere una applicazione capace di usare SSL è sufficiente inserire nel codice le chiamate a tali funzioni di libreria



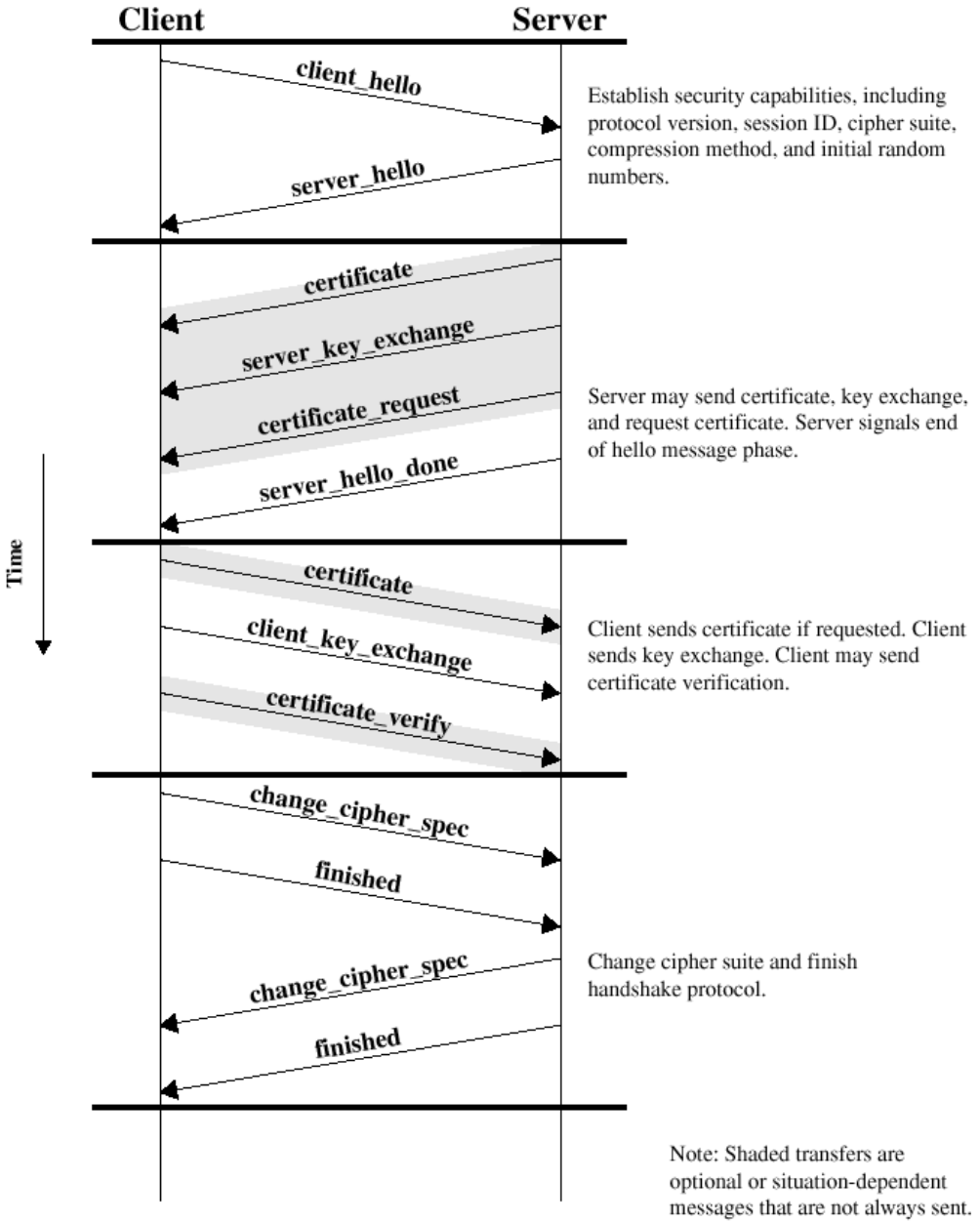
Sessione e connessione

- Due entità che colloquiano utilizzando SSL devono avere aperto una sessione
- Una singola sessione può includere numerose connessioni sicure contemporanee
- Due entità possono avere attive numerose sessioni contemporanee

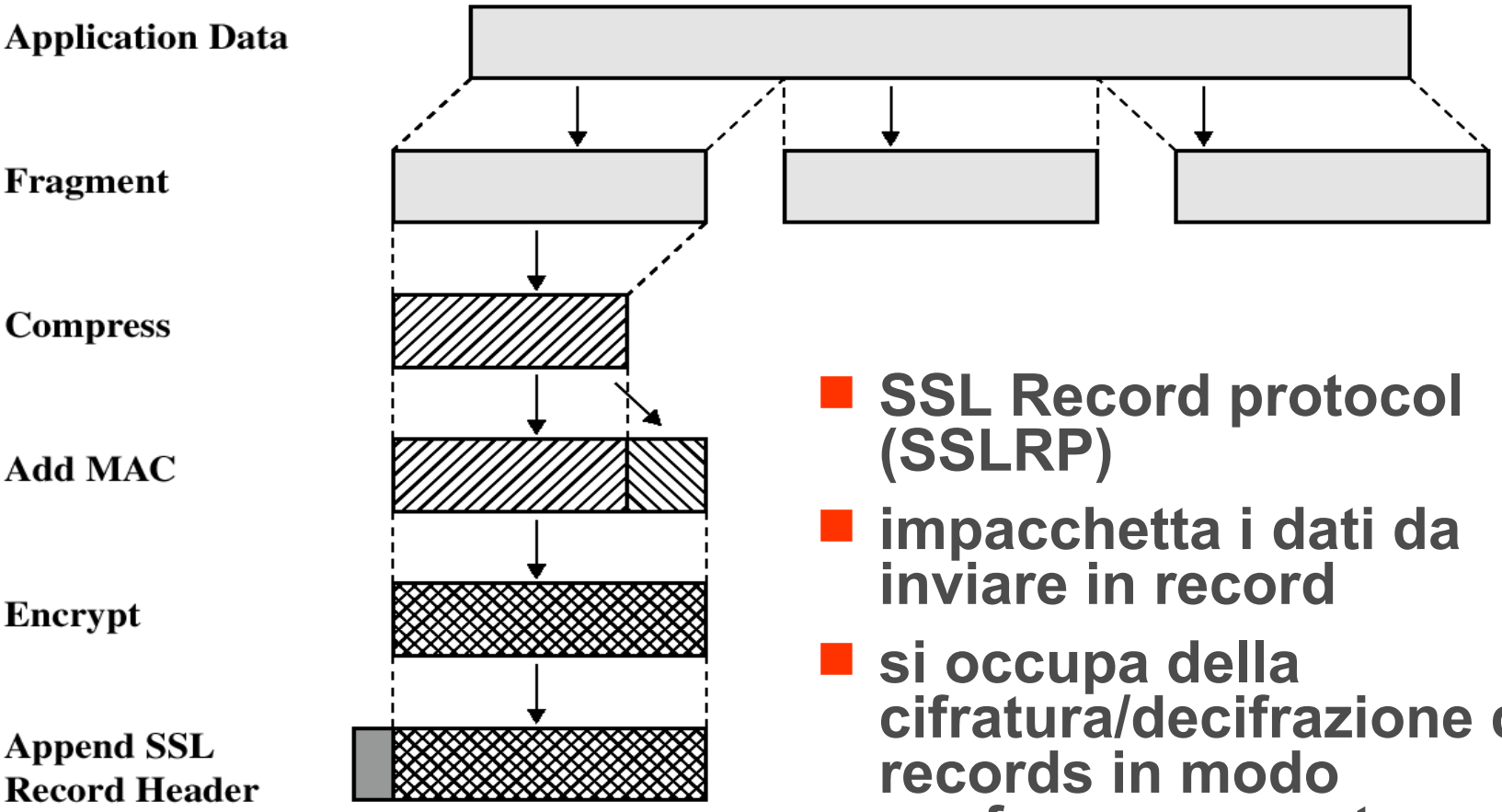


Architettura di SSL – Handshake Protocol

- La parte più complessa di SSL.
- Consente al server ed al client di autenticarsi reciprocamente
  - challenge response basato su crittografia asimmetrica e certificati X.509
  - nelle applicazioni web è comune che solo il server provi la sua autenticità al client
- Negozia gli algoritmi e le chiavi per la cifratura ed i controlli di integrità
- Interviene prima che qualsiasi dato sia trasmesso
- Progettato per limitare il carico di elaborazione e di rete
  - Implementa un meccanismo di chaching delle sessioni per limitare il numero di aperture e quindi il carico di elaborazione
  - Tenta di ridurre al minimo l'attività sulla rete (scambio di messaggi)
  - E' possibile negoziare un numero di sessione: se la comunicazione si interrompe temporaneamente, vengono poi recuperati i parametri della sessione senza rinegoziarli



Architettura di SSL – record protocol

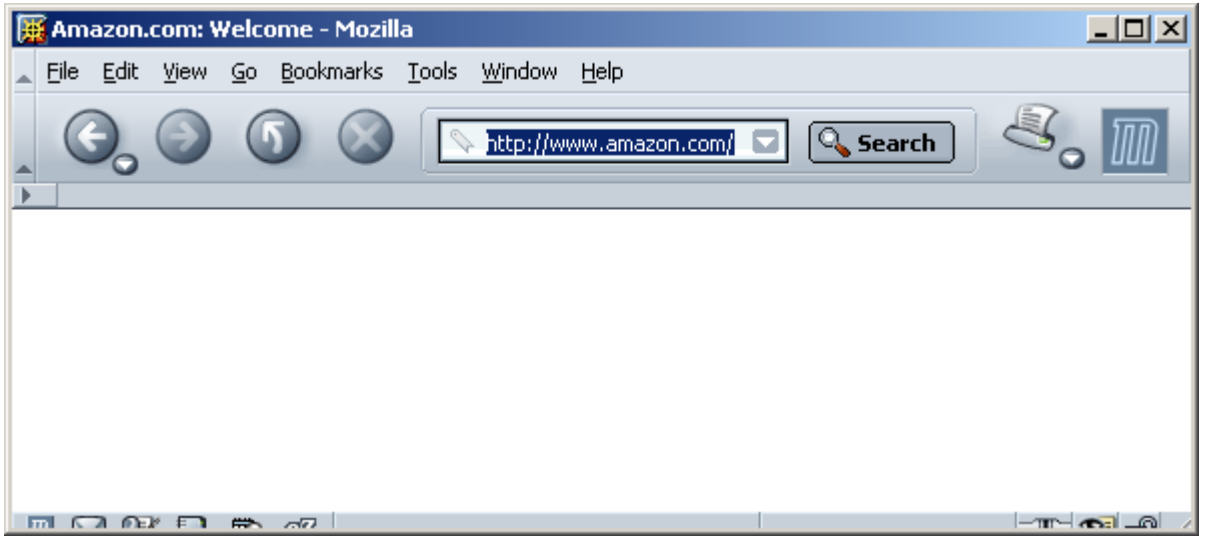


- SSL Record protocol (SSLRP)
- impacchetta i dati da inviare in record
- si occupa della cifratura/decifrazione dei records in modo conforme a quanto negoziato

SSL → TLS

- Transport Layer Security è l'evoluzione di SSL
  - Lo stesso formato di record
  - Definito in RFC 5246 (v1.2) e 8446 (v1.3)
- Simile a SSLv3, ma differenziato in:
  - numero della versione
  - codice di autenticazione del messaggio
  - funzione pseudocasuale
  - codici di avviso
  - suite di cifratura
  - tipi di certificato client
  - certificate_verify e messaggio finito
  - calcoli crittografici
  - padding

HTTPS

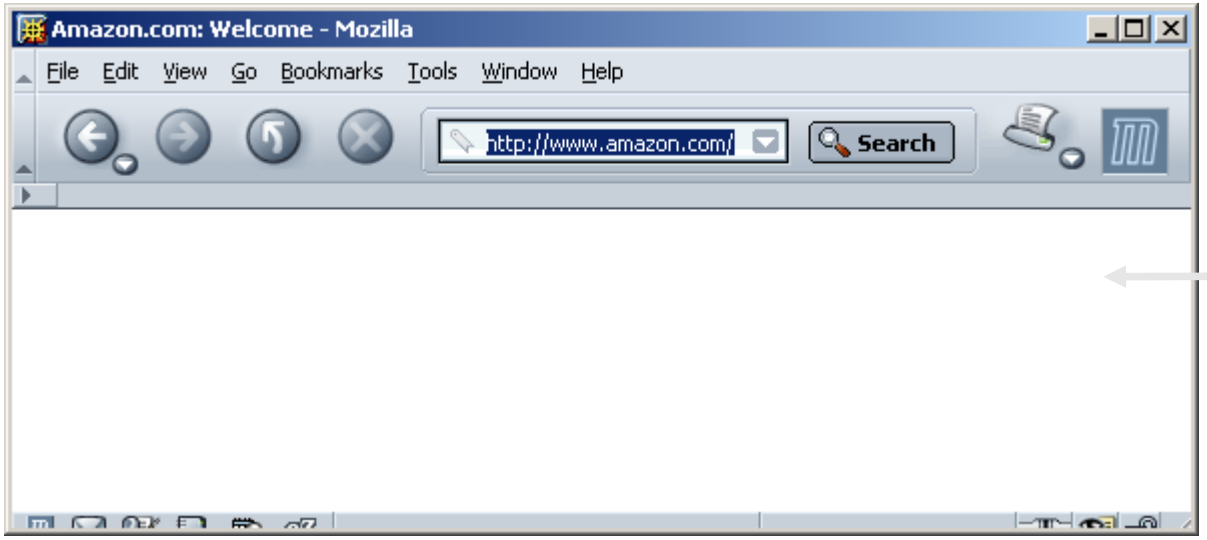


"Provami che hai la chiave privata di amazon"

amazon.com.

Web server @ crackeraddress

HTTPS



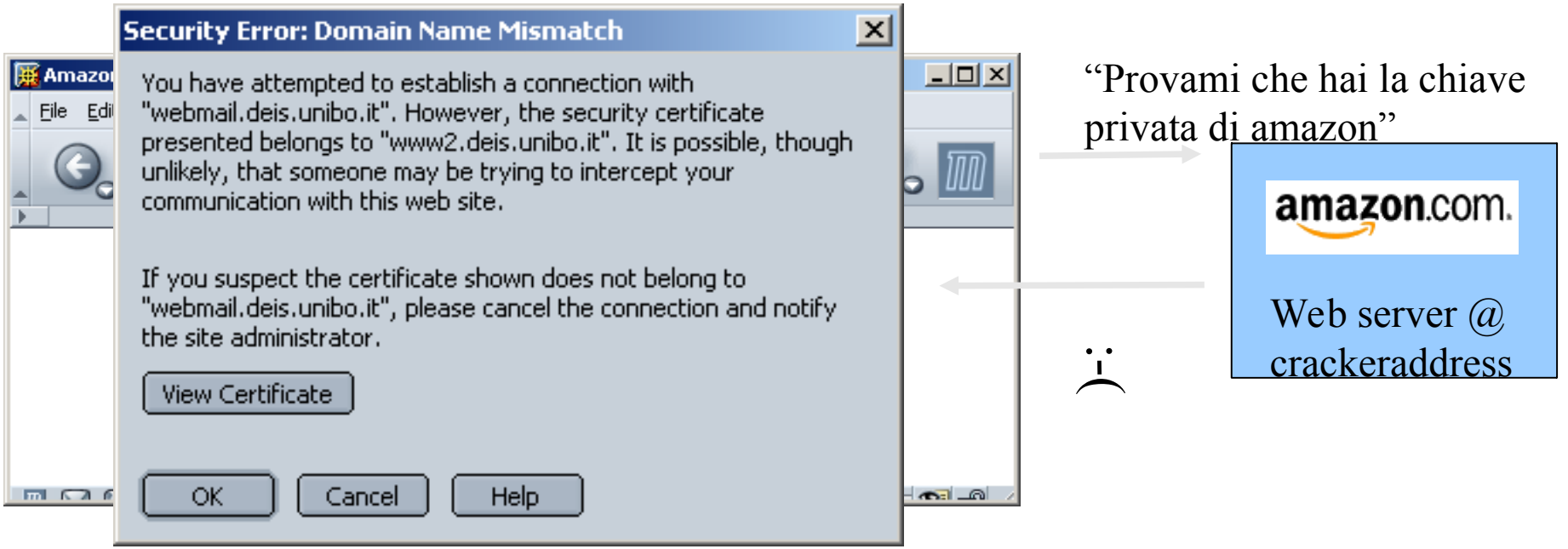
"Provami che hai la chiave privata di amazon"

amazon.com

Web server @ crackeraddress

☹

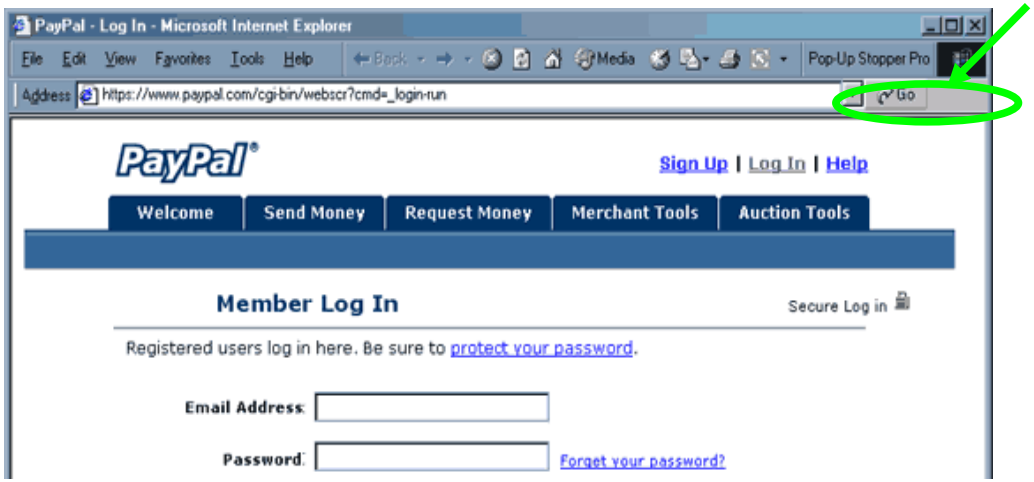
HTTPS



- **Come fa il browser a verificare la prova fornita dal web server?**
  - **Certificate store**
  - **Trusted CAs**

Occultamento della barra degli indirizzi

Il vecchio: qualche riga di codice js o activeX

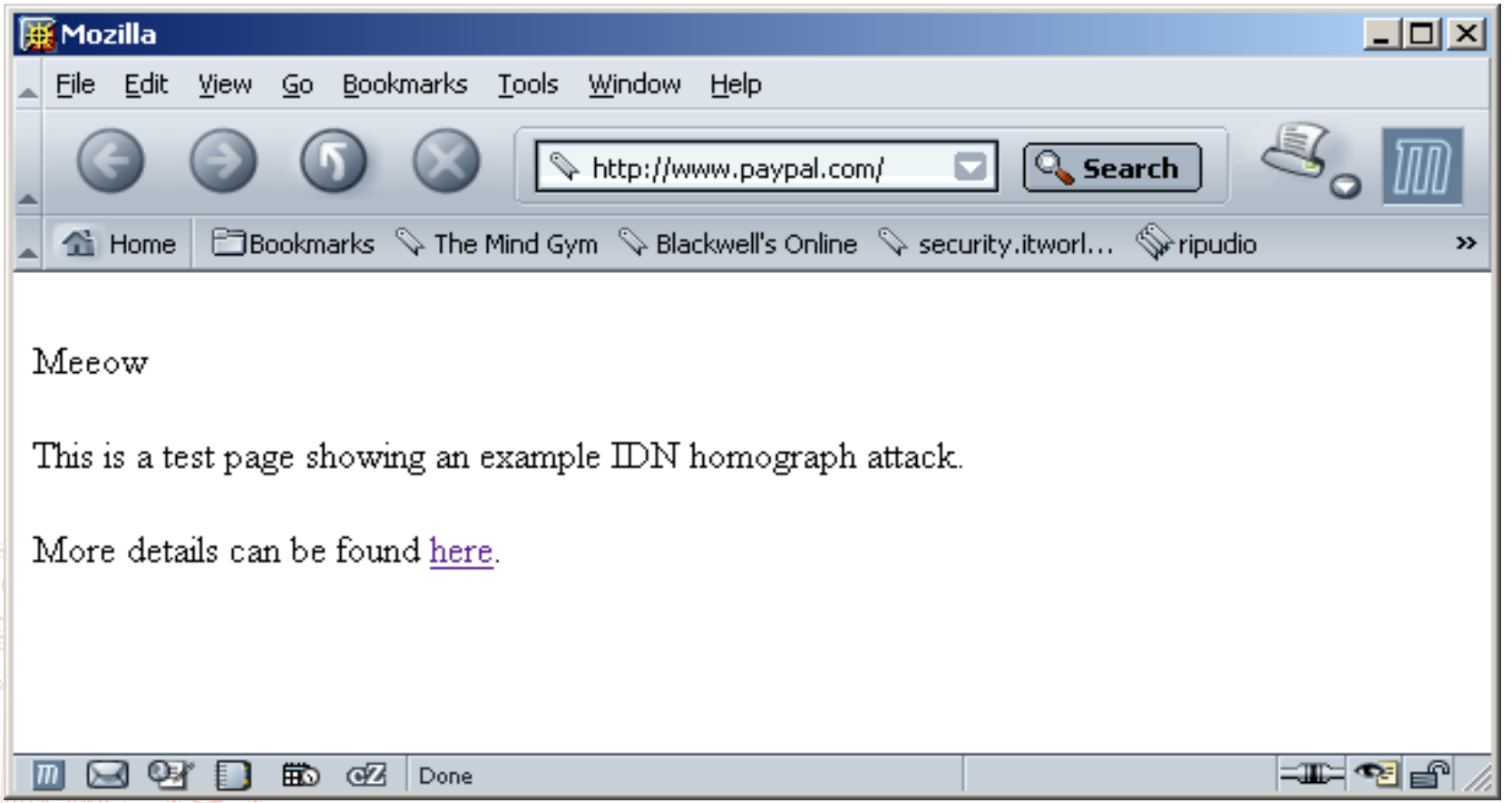


Il nuovo: auto-hiding della barra nei browser mobili

https://jameshfisher.com/2019/04/27/the-inception-bar-a-new-phishing-method/

Occultamento dell'URL

RFC3490/1/2: International Domain Names e attacci omografici

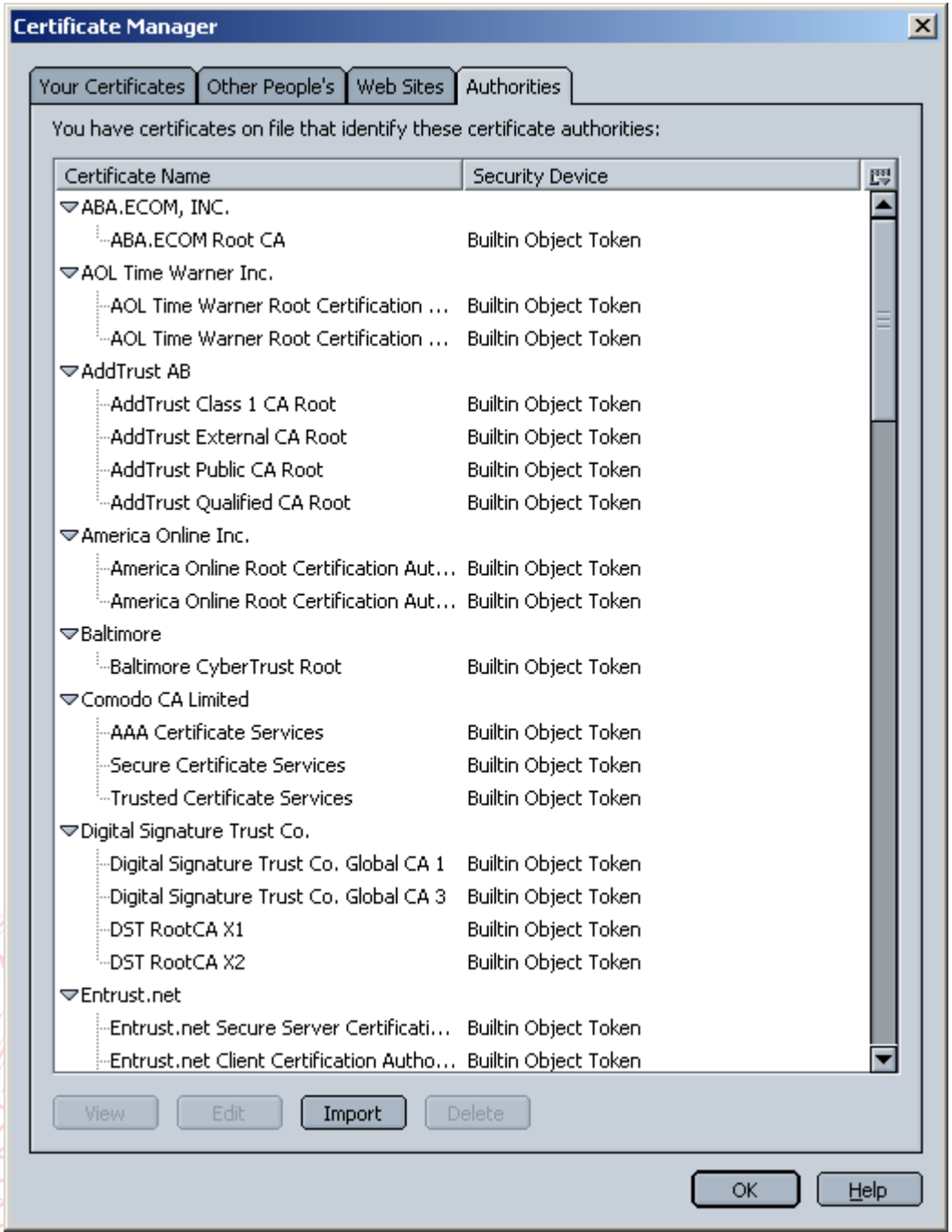


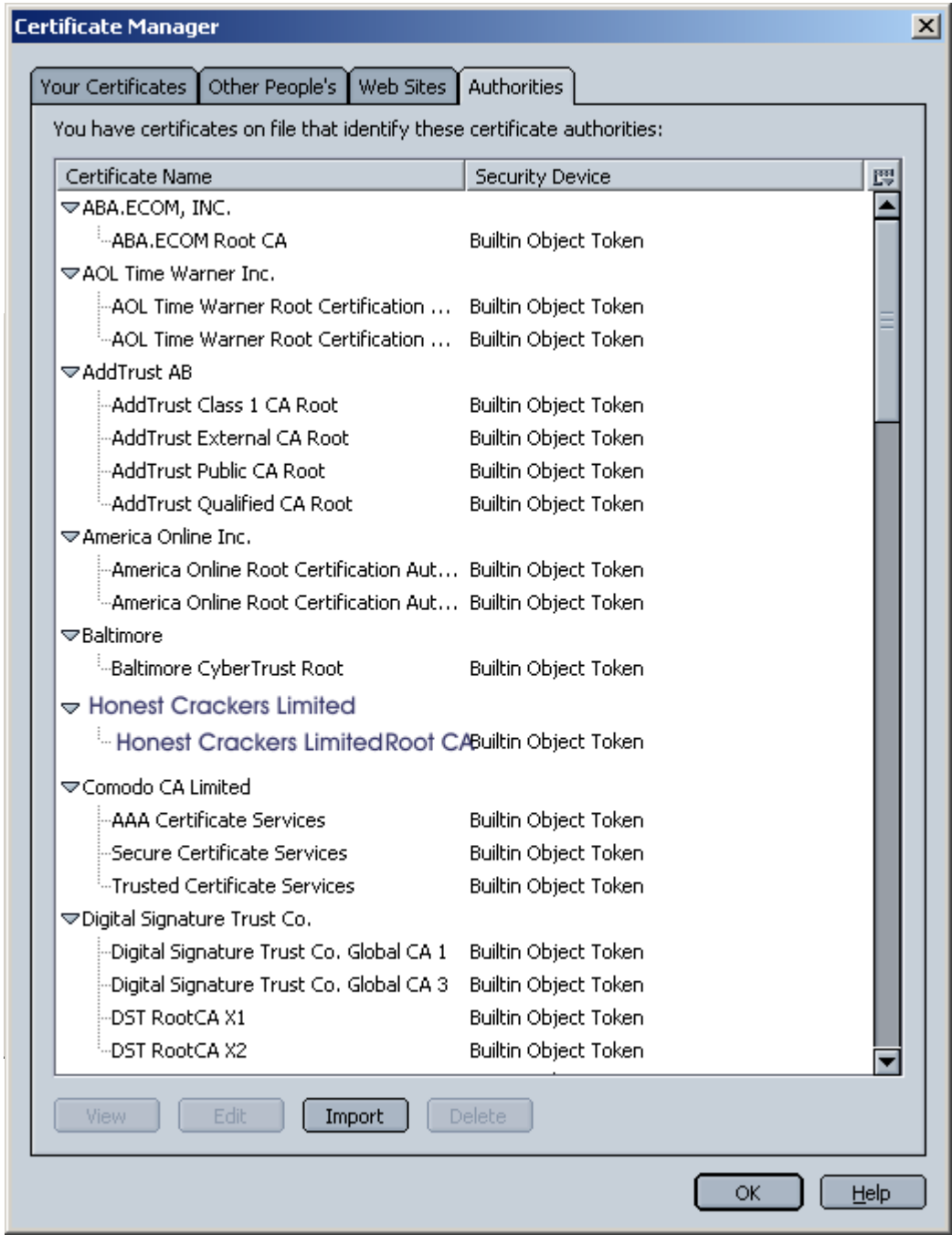
Attacchi omografici

- Problema strutturale
  - IDN consente nomi di dominio in alfabeti diversi dal latino di base
  - Il sistema DNS supporta solo il latino di base
- Soluzione: punycode
  - conversione dei caratteri internazionali in sequenze di caratteri base
    - es: 點看 → xn--c1yn36f
  - ma ci sono caratteri identici (omografi) a quelli latini
    - es: paypal.com → xn--pypal-4ve.com
- Non si può impedire a un attaccante
  - di registrare il dominio xn--pypal-4ve.com
    - nessuna necessità di DNS hijacking
  - di ottenere un legittimo certificato X.509 a tale nome
    - corretto funzionamento di HTTPS
- Soluzione più comune lato browser: visualizzare il punycode invece del font internazionale che potrebbe trarre in inganno

questa è una lettera in cirillico!

Iniezione di CA nel certificate store





Fake certificates

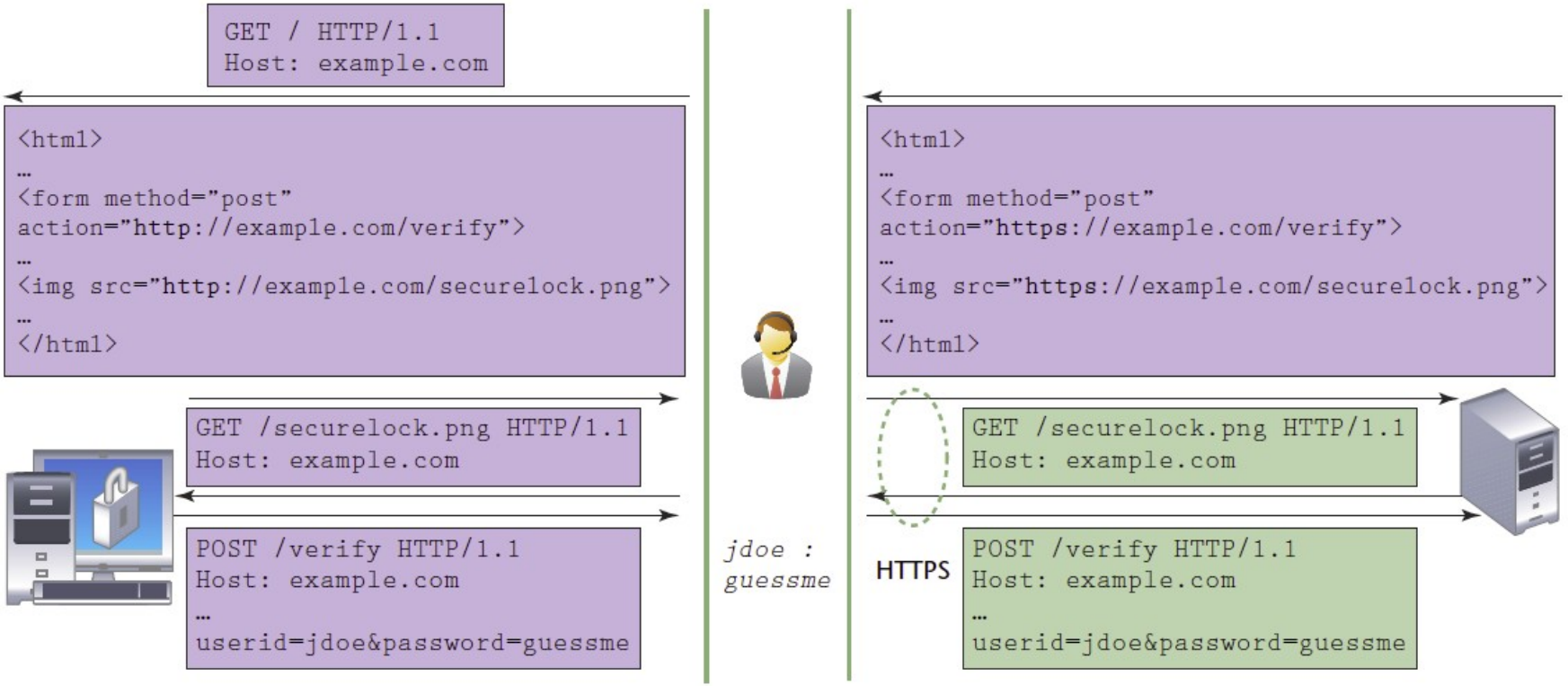
- Il modo più "semplice" di impersonare un sito è falsificare il certificato

  https://www.thesslstore.com/blog/what-is-a-rogue-certificate/

  - raro ma pericolosissimo: compromissione di una CA
  - se non emesso da un'autorità riconosciuta servono vari sotterfugi (vedi seguito)
- *HTTP Public Key Pinning (HPKP - RFC7469)*
  - limita le chiavi associabili a un dominio
    - root, intermediate, o end-entity
  - dichiarato nell'header HTTP
    - `Public-Key-Pins`
    - `Public-Key-Pins-Report-Only`
  - deprecato dal team di Google Chrome
- *Certificate Transparency (CT – RFC 6962)*
  - un framework aperto per scrutinare il processo di rilascio dei certificati; dal sito ufficiale di Google:
    - Make it impossible (or at least very difficult) for a CA to issue a SSL certificate for a domain without the certificate being visible to the owner of that domain.
    - Provide an open auditing and monitoring system that lets any domain owner or CA determine whether certificates have been mistakenly or maliciously issued.
    - Protect users (as much as possible) from being duped by certificates that were mistakenly or maliciously issued.

Stripping

- pagine HTTP che inviano dati sensibili via HTTPS possono essere modificate da un MITM



Mitigazione

- HSTS (HTTP Strict Transport Security)
  - policy implementata dai server per forzare i browser a interagire via HTTPS
  - RFC 6797
- Campo nell'header della risposta
  - inefficace sulla prima richiesta!

- Attacchi simili per i quali non c'è un equivalente di HSTS: STARTTLS command injection
  - connessioni che partono insicure e chiedono l'upgrade
  - MITM interferisce o accoda comandi in ordine errato

Vulnerabilità di SSL – a livello di protocollo

- Crittografia debole
  - uso di cifrari con problemi noti (RC4)
  - uso di IV prevedibili (BEAST)
- RC4 – non abilitatelo!
  - https://www.usenix.org/conference/usenixsecurity13/security-rc4-tls
  - http://www.crypto.com/papers/others/rc4_ksaproc.pdf
  - http://saluc.engr.uconn.edu/refs/stream_cipher/mantin01attackRC4.pdf
  - http://dblp.uni-trier.de/db/conf/sacrypt/sacrypt2007.html#PaulM07
- BEAST (2011) - Browser Exploit Against SSL/TLS
  - https://nvd.nist.gov/vuln/detail/CVE-2011-3389

    "The SSL protocol, as used in certain configurations in Microsoft Windows and Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Opera, and other products, encrypts data by using CBC mode with chained initialization vectors, which allows man-in-the-middle attackers to obtain plaintext HTTP headers via a blockwise chosen-boundary attack (BCBA) on an HTTPS session, in conjunction with JavaScript code that uses (1) the HTML5 WebSocket API, (2) the Java URLConnection API, or (3) the Silverlight WebClient API, aka a "BEAST" attack."

Vulnerabilità di SSL – a livello di protocollo

- Padding Oracle Attacks
  - https://www.iacr.org/cryptodb/archive/2002/EUROCRYPT/2850/2850.pdf
  - causato dalla sequenza MAC-then-encrypt
  - Lucky Thirteen
  - Padding Oracle On Downgraded Legacy Encryption (POODLE)

Vulnerabilità di SSL – a livello di protocollo

- Lucky Thirteen (2013)
  - https://nvd.nist.gov/vuln/detail/CVE-2013-0169

    "The TLS protocol 1.1 and 1.2 and the DTLS protocol 1.0 and 1.2, as used in OpenSSL, OpenJDK, PolarSSL, and other products, do not properly consider timing side-channel attacks on a MAC check requirement during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets"
  - basso impatto
  - mitigato da scelte crittografiche più accorte
    - https://tools.ietf.org/html/rfc5288
    - https://tools.ietf.org/html/rfc7366

Vulnerabilità di SSL – a livello di protocollo

- POODLE (2014)
  - https://nvd.nist.gov/vuln/detail/CVE-2014-3566
  - https://www.openssl.org/~bodo/ssl-poodle.pdf
  - Un attaccante in grado di posizionarsi *in the middle* (ad esempio contro gli utenti di un hotspot pubblico) può forzare il downgrade delle connessioni verso SSLv3
  - SSLv3 usa cifrari deboli
    - RC4 per stream
    - block ciphers con CBC vulnerabile a padding oracle
  - Impatto: decifrazione traffico
  - unica mitigazione disabilitare SSLv3

Vulnerabilità di SSL – a livello di protocollo

- Legati alla compressione
  - CRIME
  - TIME
  - BREACH

Vulnerabilità di SSL – a livello di protocollo

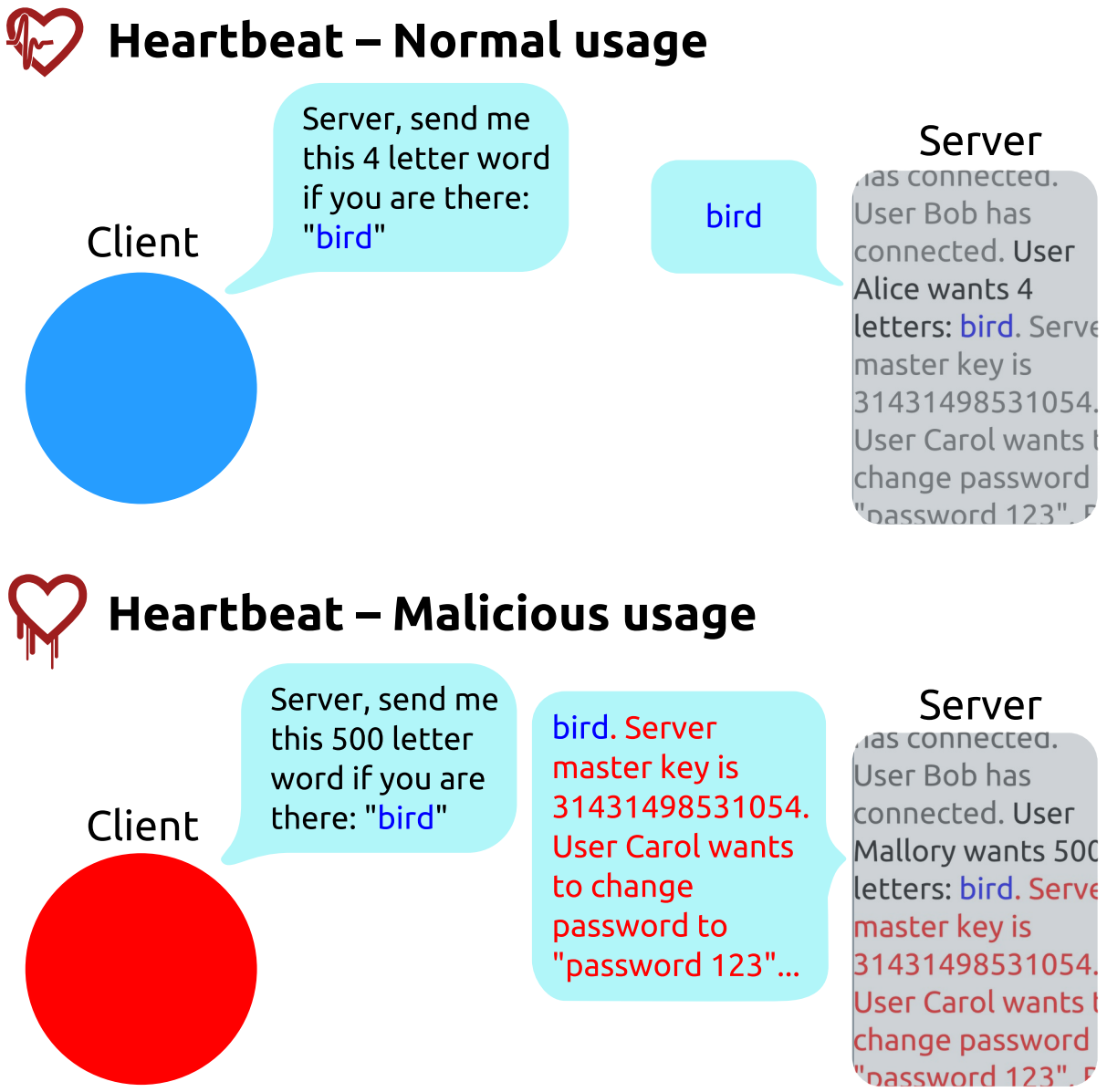
- DROWN (2016) - Decrypting RSA with Obsolete and Weakened eNcryption
  - https://drownattack.com/
  - https://nvd.nist.gov/vuln/detail/CVE-2016-0800
  - Un altro attacco reso possibile dal supporto a vecchie versioni di SSL
  - Gravi vulnerabilità note nella vecchia versione SSLv2, originata dalle restrizioni imposte dal governo USA all'esportazione di crittografia forte
    - Possibile inviare probe che limitano lo spazio di ricerca delle chiavi a 40 bit
  - Se tale versione è supportata su un server con una certa chiave privata, tutti i server che usano tale chiave sono vulnerabili
  - Impatto: controllo completo, impersonamento del server

Vulnerabilità di SSL – a livello di implementazione

- Heartbleed (2014) - http://heartbleed.com/

  - Implementazione errata della heartbeat extension del protocollo, RFC 6520) nella libreria OpenSSL
    - Heartbeat: scambio di dati finalizzato solo a mantenere viva la connessione
    - Il browser manda una stringa che il server deve restituire, specificandone anche la lunghezza
    - Bug: il server utilizza la lunghezza dichiarata per accedere alla propria memoria, senza controllare la coerenza con la stringa ricevuta
  - Consente di leggere pezzi di memoria del sistema target
  - Impatto: possibile leak di materiale sensibile, come le chiavi

Heartbleed in pillole

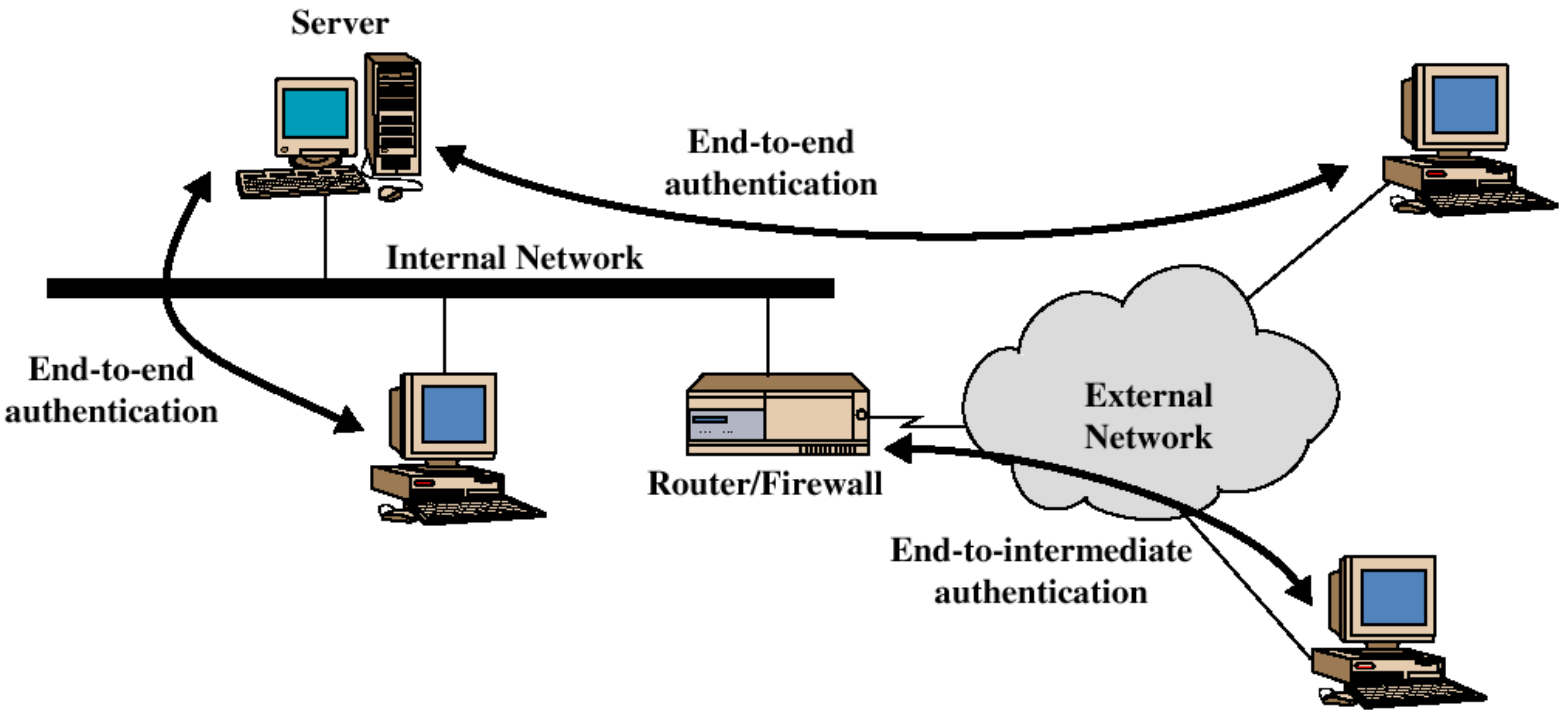


HTTP security - riferimenti

- Iniziative per evitare connessioni in chiaro
  - https://tools.ietf.org/html/rfc6797
  - https://www.eff.org/it/https-everywhere
  - https://tools.ietf.org/html/rfc8555
  - https://letsencrypt.org/
- HSTS / pinning / CT
  - https://tools.ietf.org/html/rfc7469
  - https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning
  - https://tools.ietf.org/html/rfc6962
  - https://www.certificate-transparency.org/
- address bar
  - https://en.wikipedia.org/wiki/IDN_homograph_attack
  - https://www.netsparker.com/blog/web-security/web-browser-address-bar-spoofing/
- TLS – dalle basi alle vulnerabilità
  - https://www.acunetix.com/blog/articles/tls-security-what-is-tls-ssl-part-1/
    - primo di 6 articoli
  - https://tools.ietf.org/html/rfc7457

Virtual Private Network

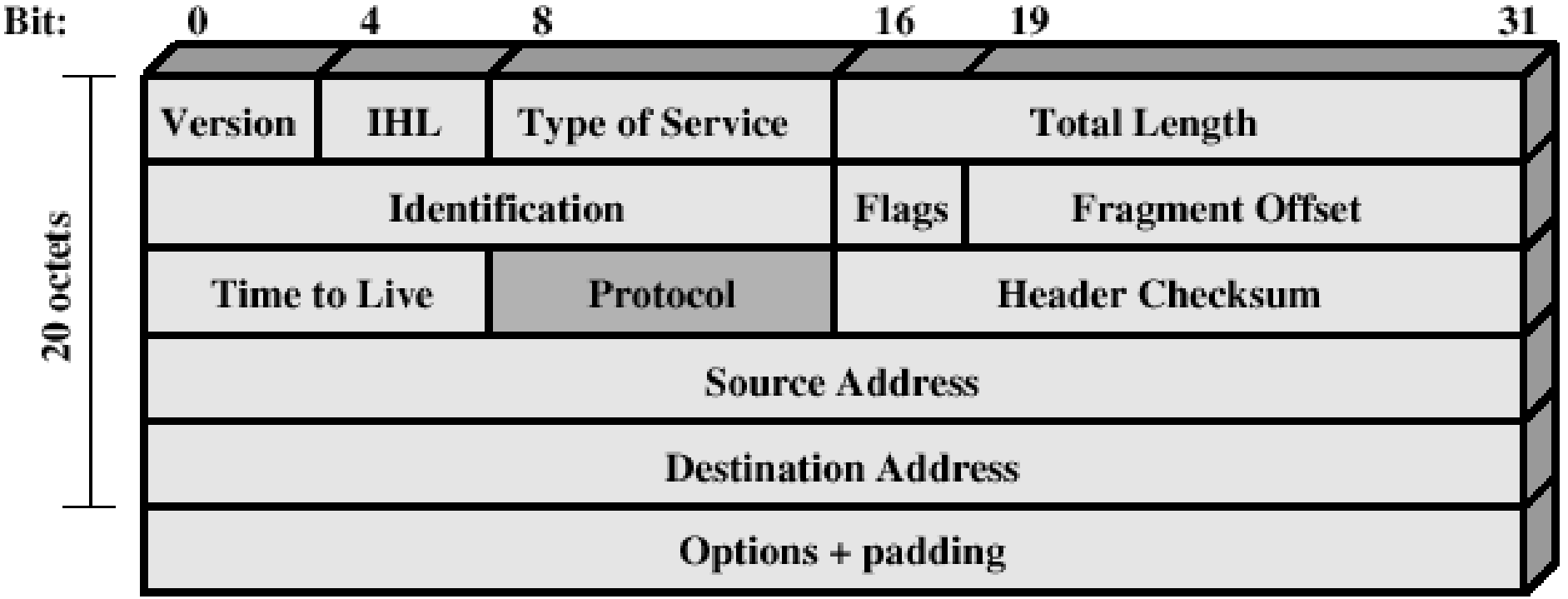
- Reti virtualmente private = metodi di trasporto del traffico
  - private perché garantiscono sicurezza
  - virtualmente perché il traffico è convogliato attraverso reti insicure
- Scenari host-host / host-net / net-net



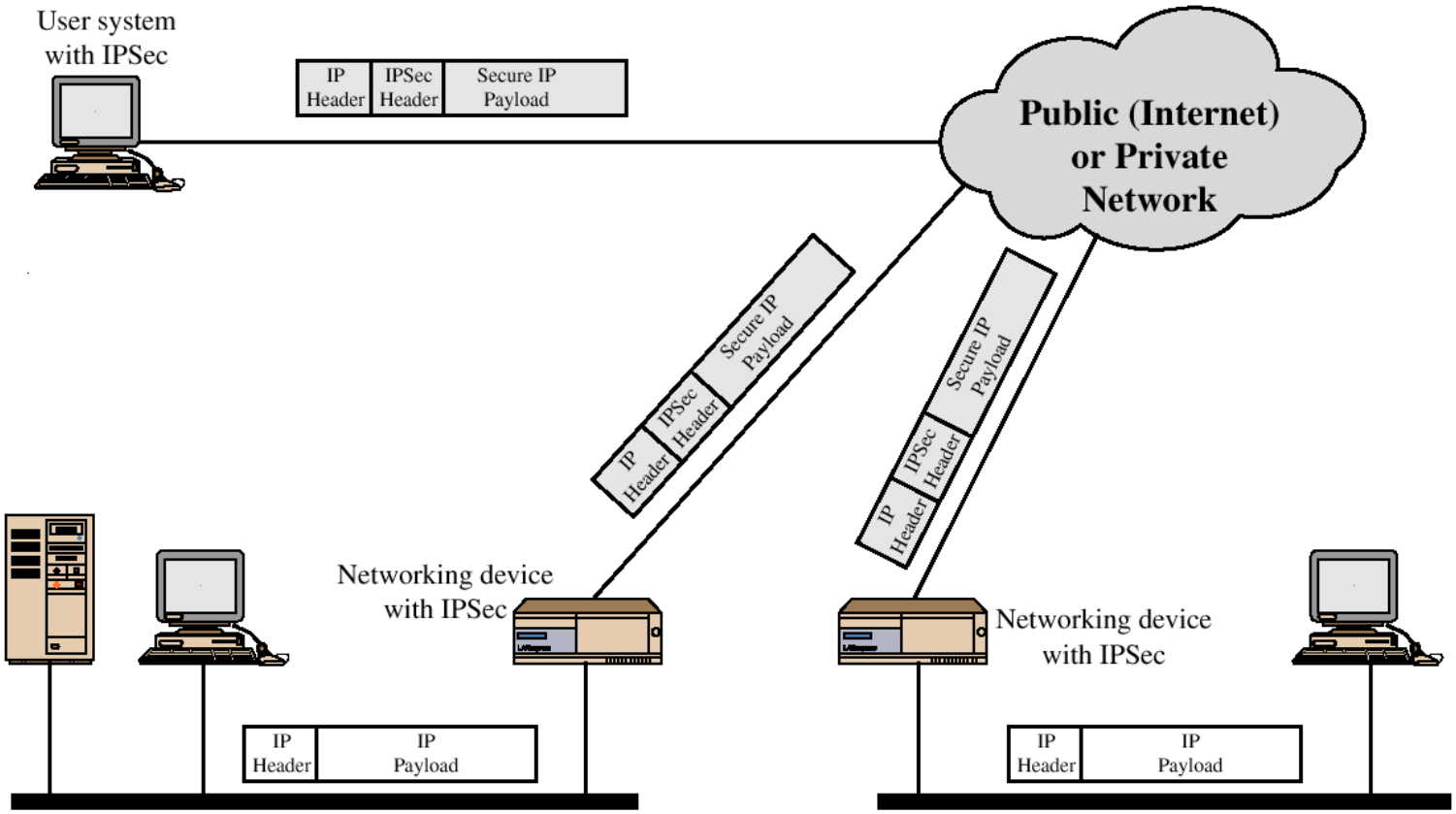
IP Security

- IPSec non è un protocollo singolo
  - set di algoritmi di sicurezza
  - framework per la negoziazione degli algoritmi
  - specifiche per la gestione delle chiavi
- Applicazioni di IPSec
  - Interconnessione di sedi remote attraverso Internet
  - Accesso di client alla rete aziendale attraverso Internet
  - Creazione di reti complesse con criteri di protezione differenziati
- Vantaggi di IPSec
  - Trasparente alle applicazioni
  - Applicabile al traffico infrastrutturale di Internet, come i messaggi che i router si scambiano per aggiornare le tabelle di instradamento

IPv4 Header

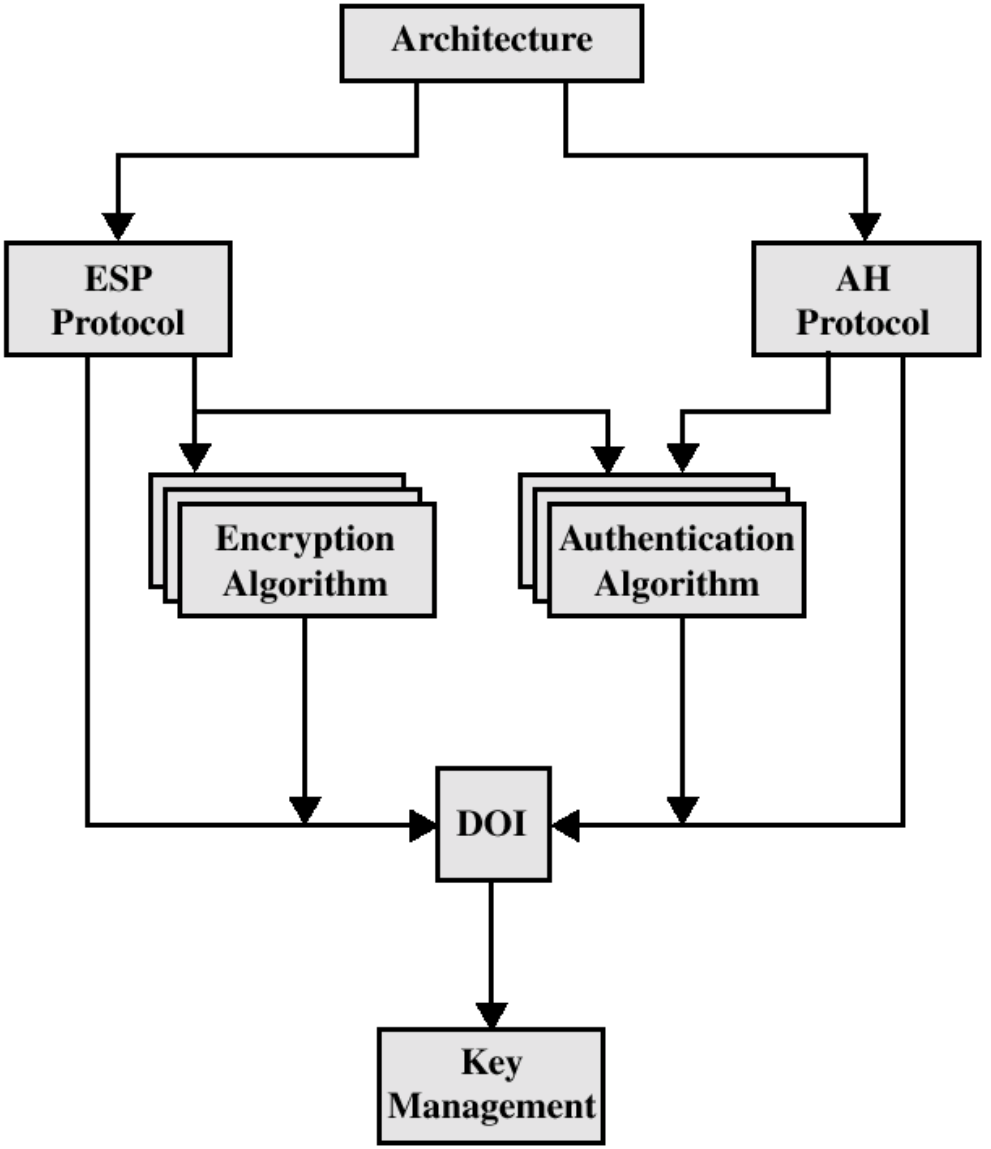

Bit:
0
4
8
16
19
31
20 octets
Version
IHL
Type of Service
Total Length
Identification
Flags
Fragment Offset
Time to Live
Protocol
Header Checksum
Source Address
Destination Address
Options + padding

IPSec – scenari



Gli standard di IPSec

- RFC 4301: Security Architecture for the Internet Protocol
  - upd: 6040, 7619
- RFC 4302: IP Authentication Header (AH)
- RFC 4303: IP Encapsulating Security Payload (ESP)
- RFC 8821: Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)
- RFC 7296: Internet Key Exchange Protocol Version 2 (IKEv2)
  - upd: 7427, 7670, 8247, 8983



Servizi offerti ed algoritmi utilizzati

- Controllo dell'accesso
- Integrità anche senza connessione
- Autenticazione dell'origine dei dati
- Rilevazione dei replay
- Riservatezza dei dati
- Parziale riservatezza dei flussi di traffico

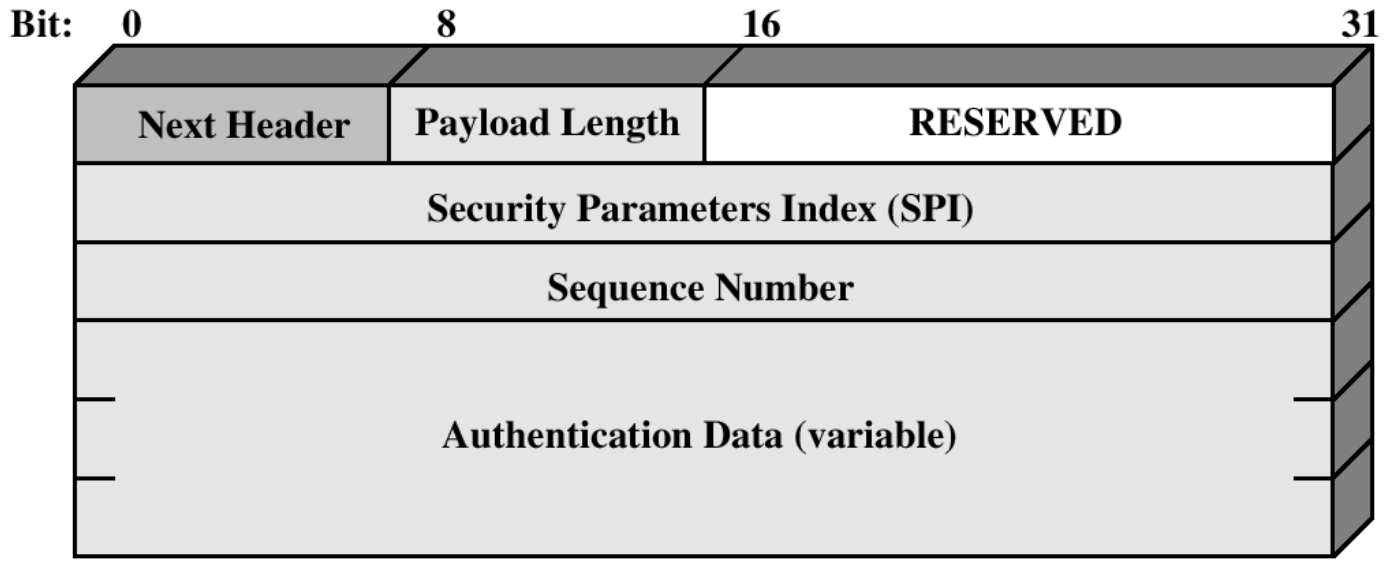
- Cifratura:
  - Three-key triple DES
  - RC5
  - IDEA
  - Three-key triple IDEA
  - CAST
  - Blowfish
- Autenticazione:
  - HMAC-MD5-96
  - HMAC-SHA-1-96
- Gestione chiavi:
  - Manuale
  - Automatizzata
    - Oakley Key Determination Protocol
    - Internet Security Association and Key Management Protocol (ISAKMP)

Terminologia di base

- SA (Security Association)
  - relazione unidirezionale tra mittente e destinatario, definita da
    - Security Parameter Index (SPI)
    - IP Destination address
    - Security Protocol Identifier
  - due modalità possibili di SA
    - Transport Mode
    - Tunnel Mode
- Protocolli di sicurezza
  - AH (Authentication Header)
  - ESP (Encapsulating Security Payload)
- Instradamento
  - Le SA vengono attivate da tabelle specializzate del sistema operativo, nel caso di Linux le extended route (eroute)

Authentication Header

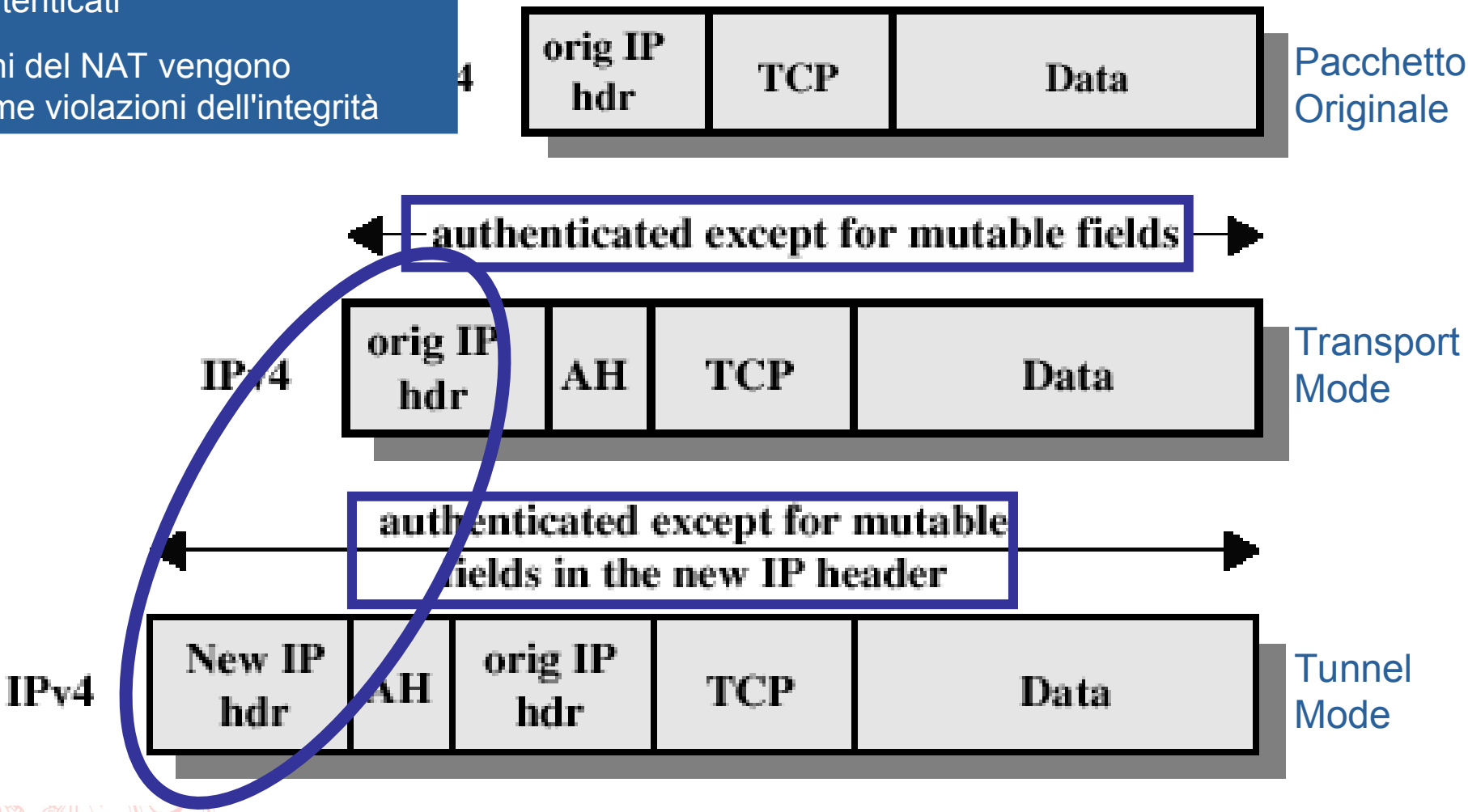
- Garantisce l'autenticazione e l'integrità dei pacchetti IP
- Protegge dai replay attacks



Autentication Header

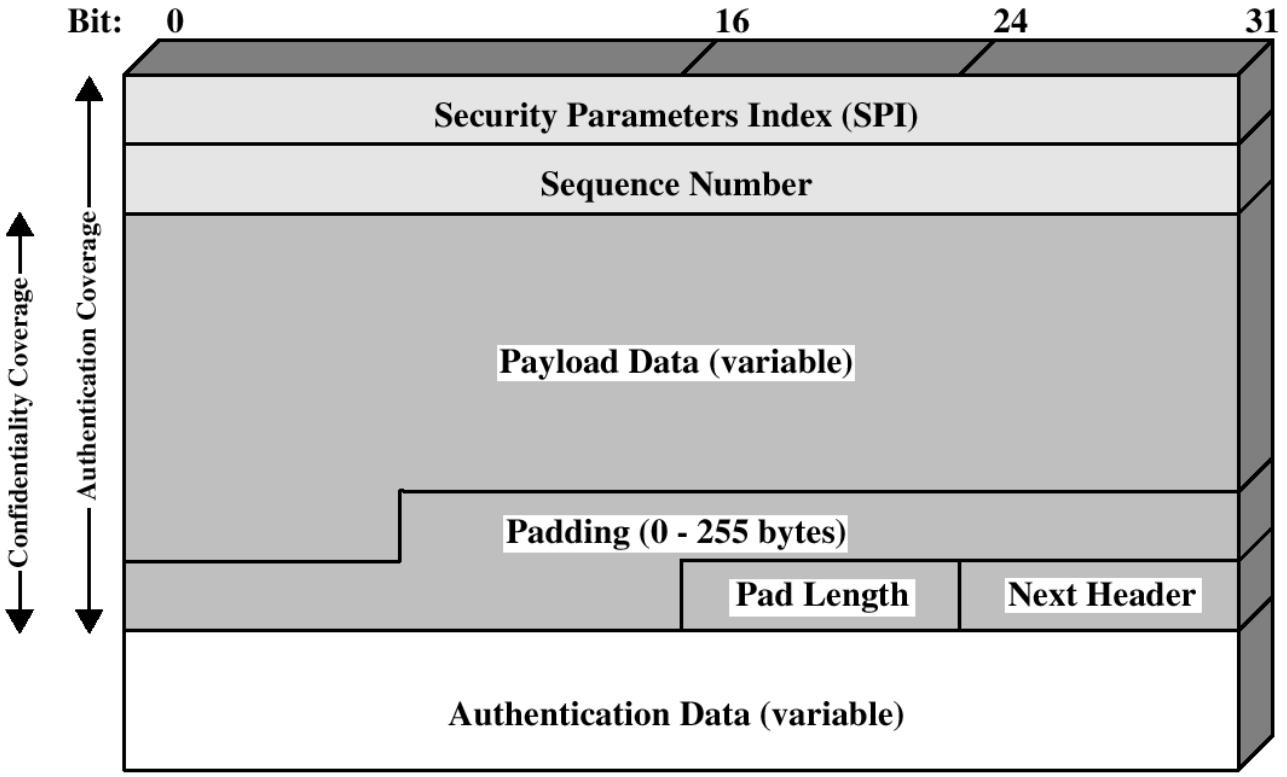
Gli indirizzi, giustamente, non sono considerati campi variabili

- vengono autenticati
- le alterazioni del NAT vengono percepite come violazioni dell'integrità



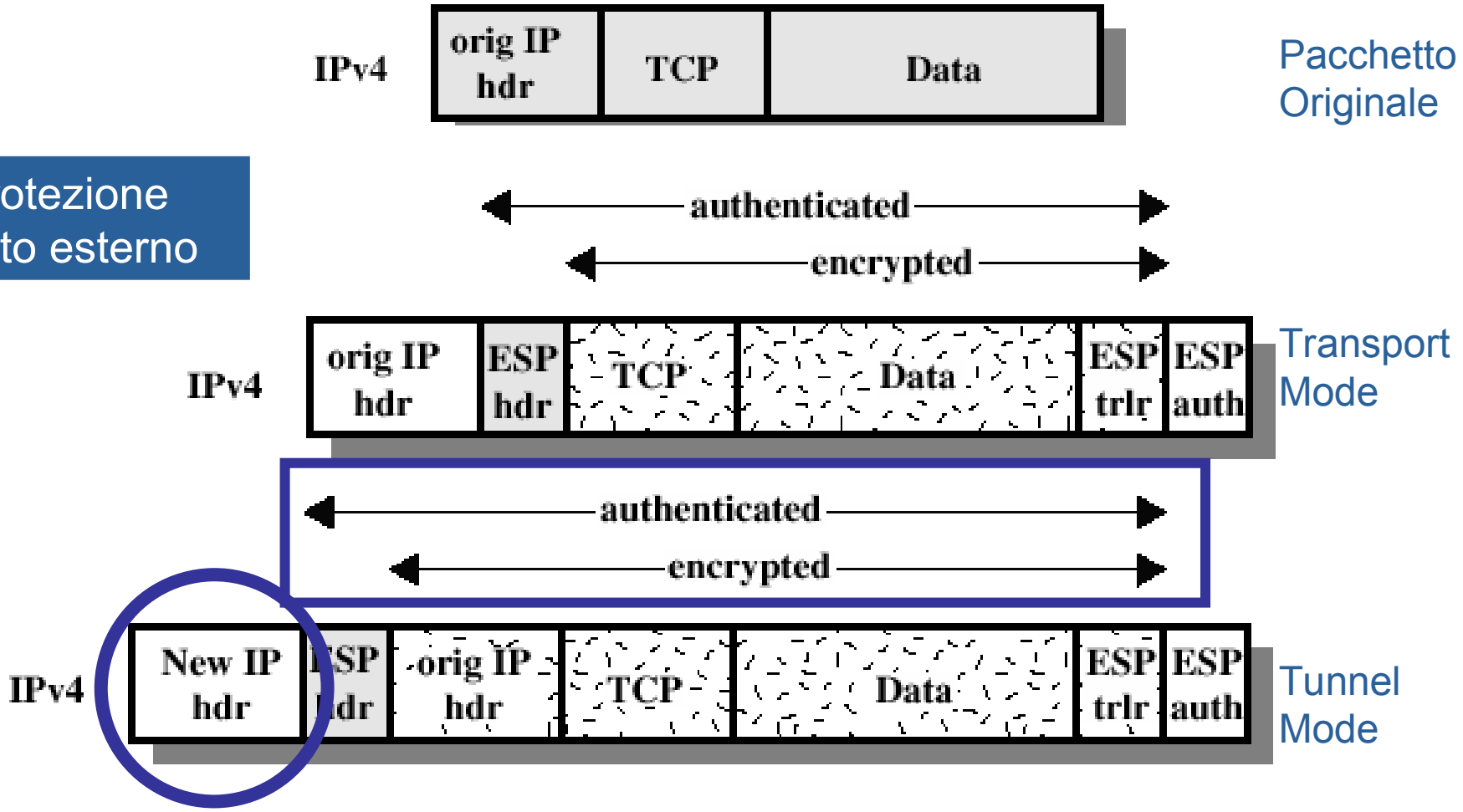
Encapsulating Security Payload

- ESP offre essenzialmente servizi per la riservatezza
- Opzionalmente anche aut/int in grado minore di AH



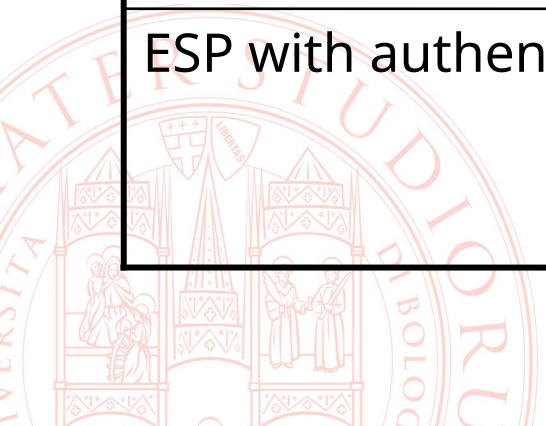
ESP con cifratura ed autenticazione

Nessuna protezione del pacchetto esterno

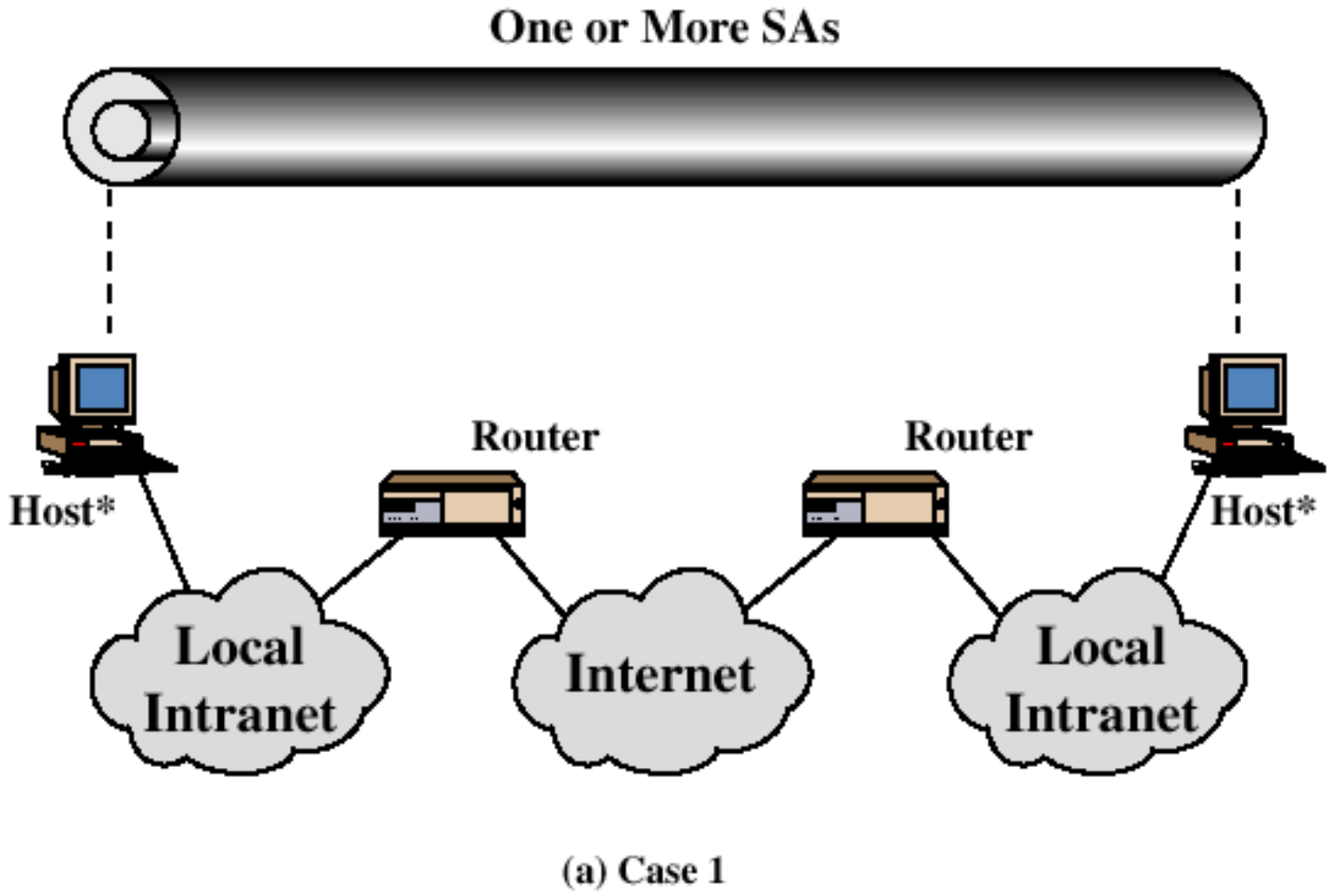


Riassunto delle combinazioni dei modi di protezione

	Transport Mode SA	Tunnel Mode SA
AH	Autentica il payload del pacchetto IP ed alcuni campi dell'header IP	Autentica l'intero pacchetto IP interno ed alcuni campi del pacchetto IP esterno
ESP	Cifra il contenuto del pacchetto	Cifra l'intero pacchetto IP interno
ESP with authentication	Cifra il contenuto del pacchetto. Autentica il payload del pacchetto ma non l'header IP	Cifra ed autentica l'intero pacchetto IP interno.

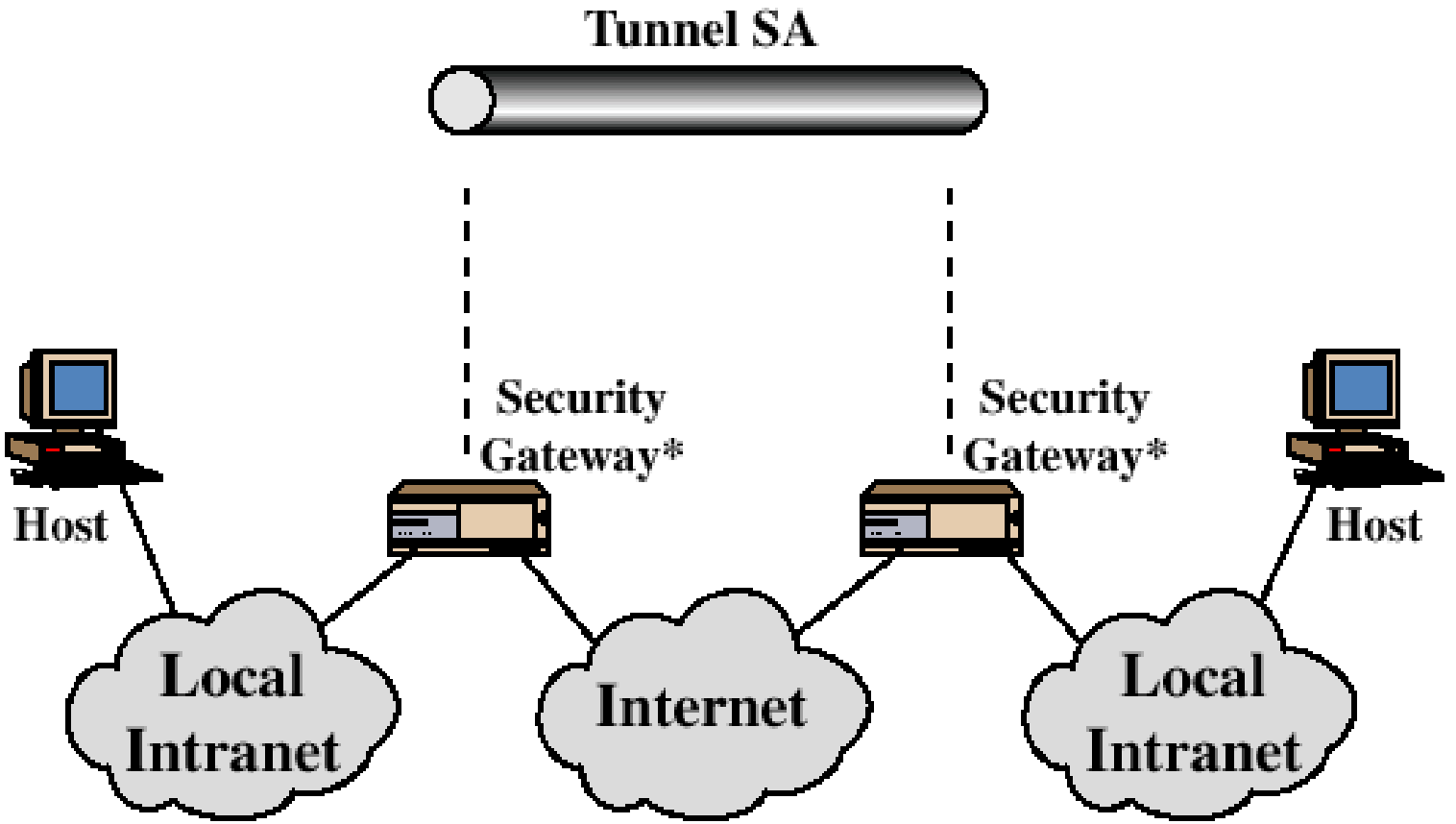


Esempi di Combinazione di SA



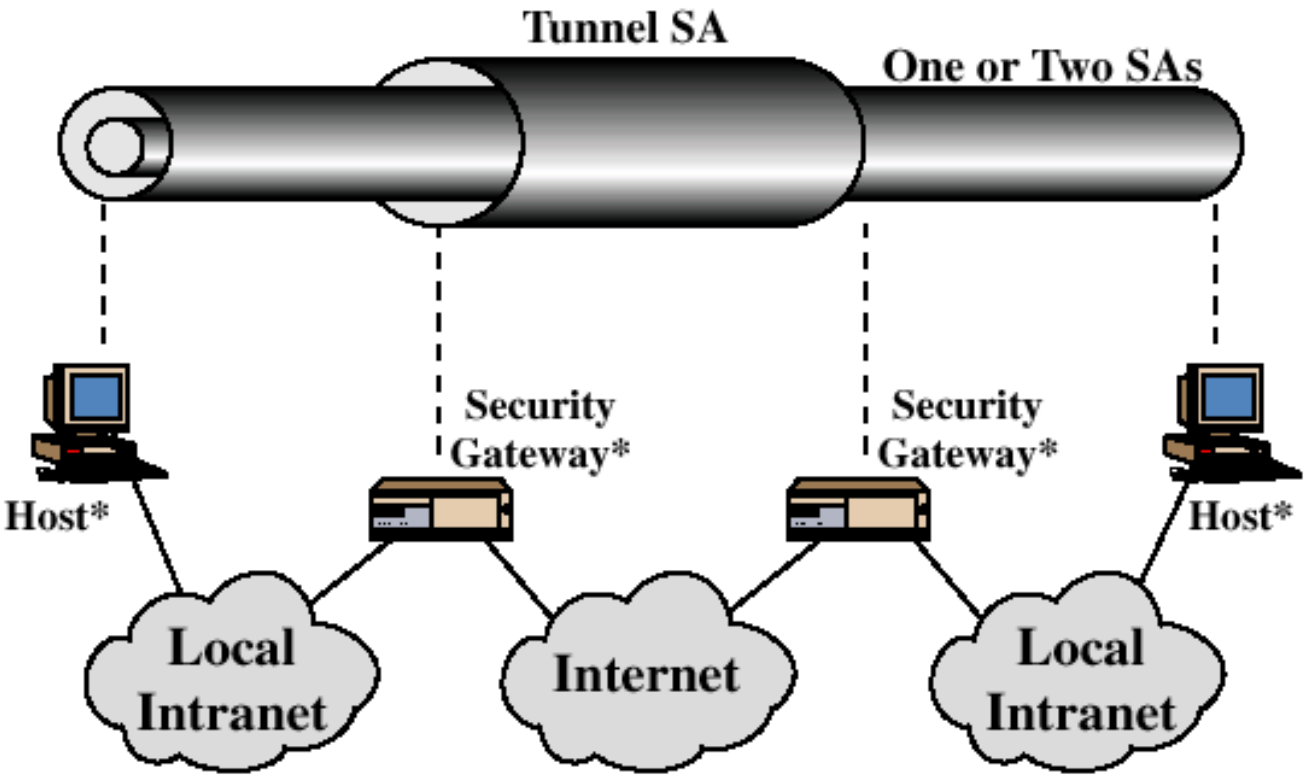
(a) Case 1

Esempi di Combinazione di SA



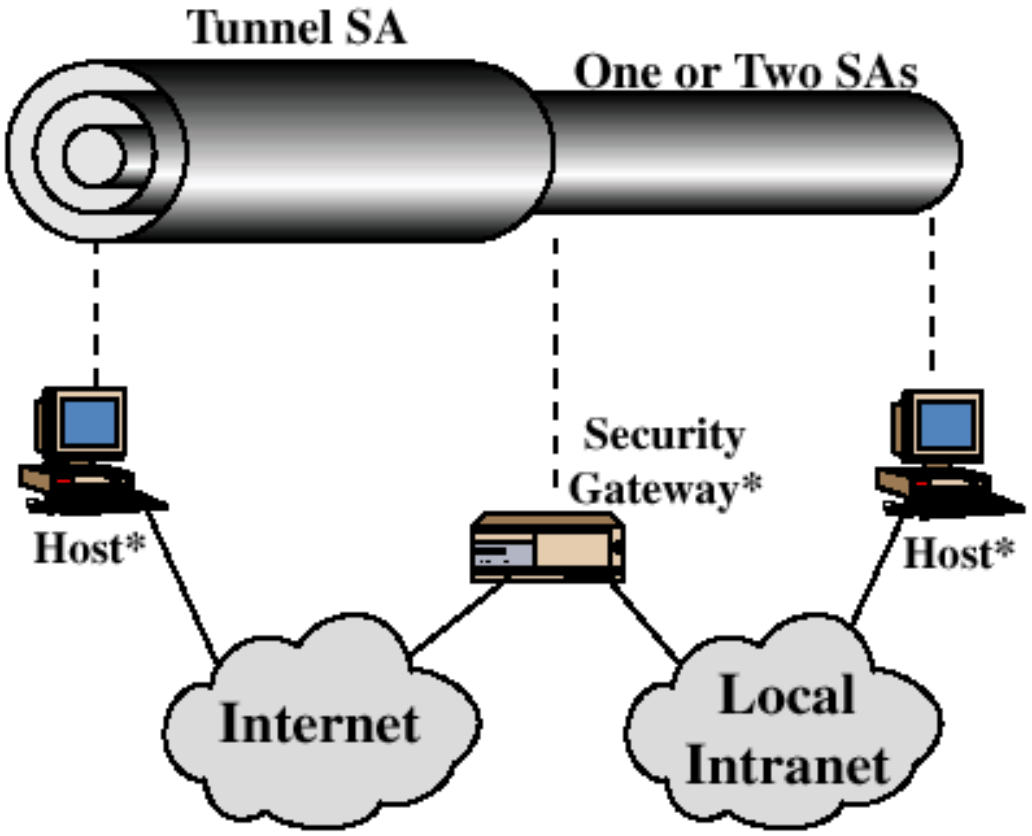
(b) Case 2

Esempi di Combinazione di SA



(c) Case 3

Esempi di Combinazione di SA



(d) Case 4

Considerazioni comparative

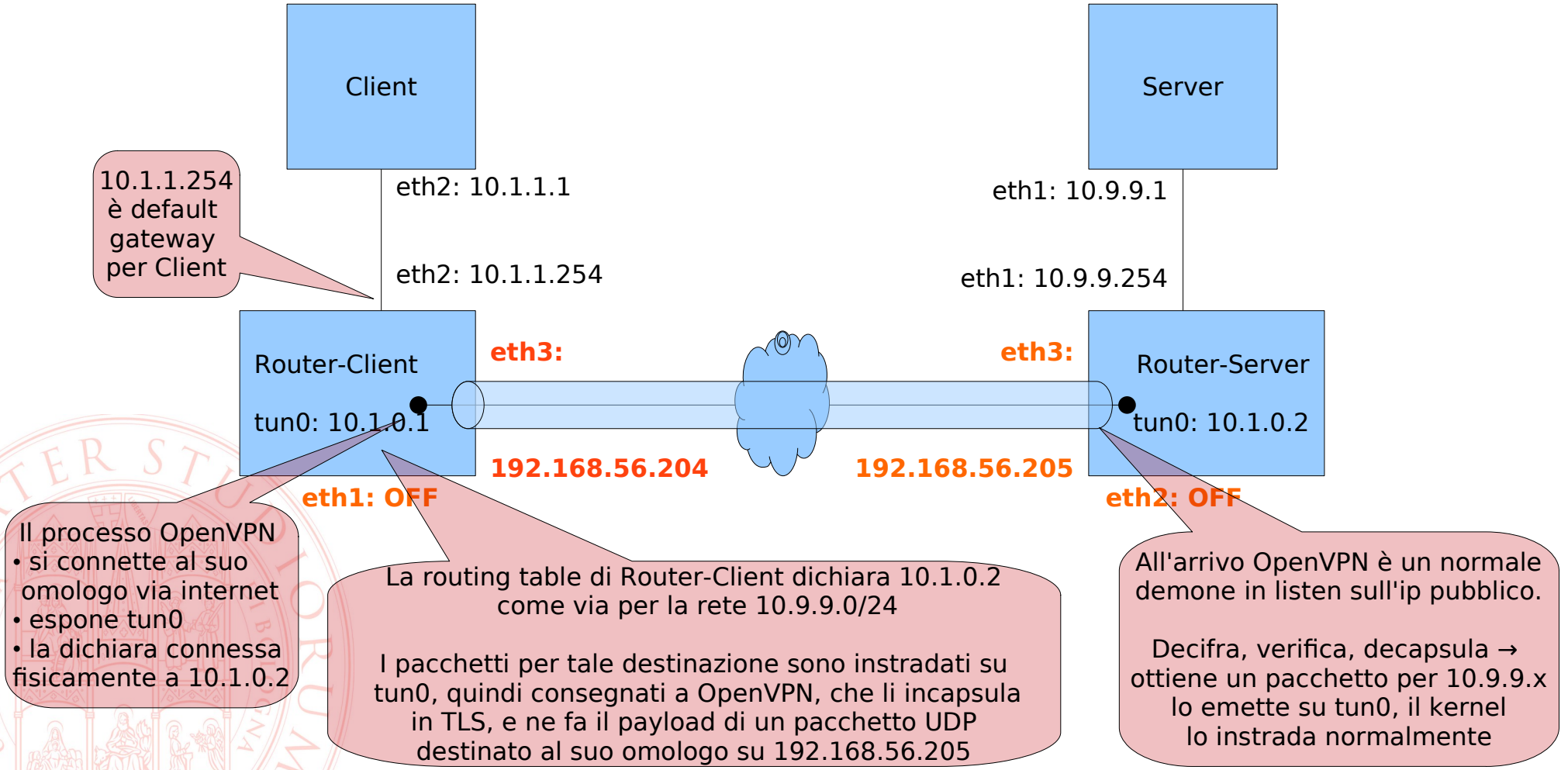
- SSL/TLS
  - è specifico di un dominio applicativo ☹
  - è semplice e realmente standard ☺
- IPSec
  - è generale e trasparente alle applicazioni ☺
  - è tipicamente implementato nello stack TCP/IP del sistema operativo, con variazioni che rendono difficile l'interoperabilità ☹
- Soluzioni "ibride"
  - utilizzo di varianti di SSL per il trasporto di pacchetti IP analogo al tunnel mode di IPSec
  - implementazione user space, indipendente dal S.O.
    - Es: OpenVPN

OpenVPN

- OpenVPN riproduce con software in user space i concetti di transport e tunnel mode di IPSec
- Serve comunque un piccolo componente kernel space: la generazione di interfacce di rete virtuali, rispettivamente di tipo *tap* e *tun*
  - queste interfacce si usano esattamente come quelle reali
  - i pacchetti inviati a un'interfaccia reale sono inviate al device driver della scheda hardware
  - i pacchetti inviati a un'interfaccia virtuale sono inviati al processo che le ha create
  - l'uso o meno di queste interfacce è determinato da normali entry nella routing table dell'host

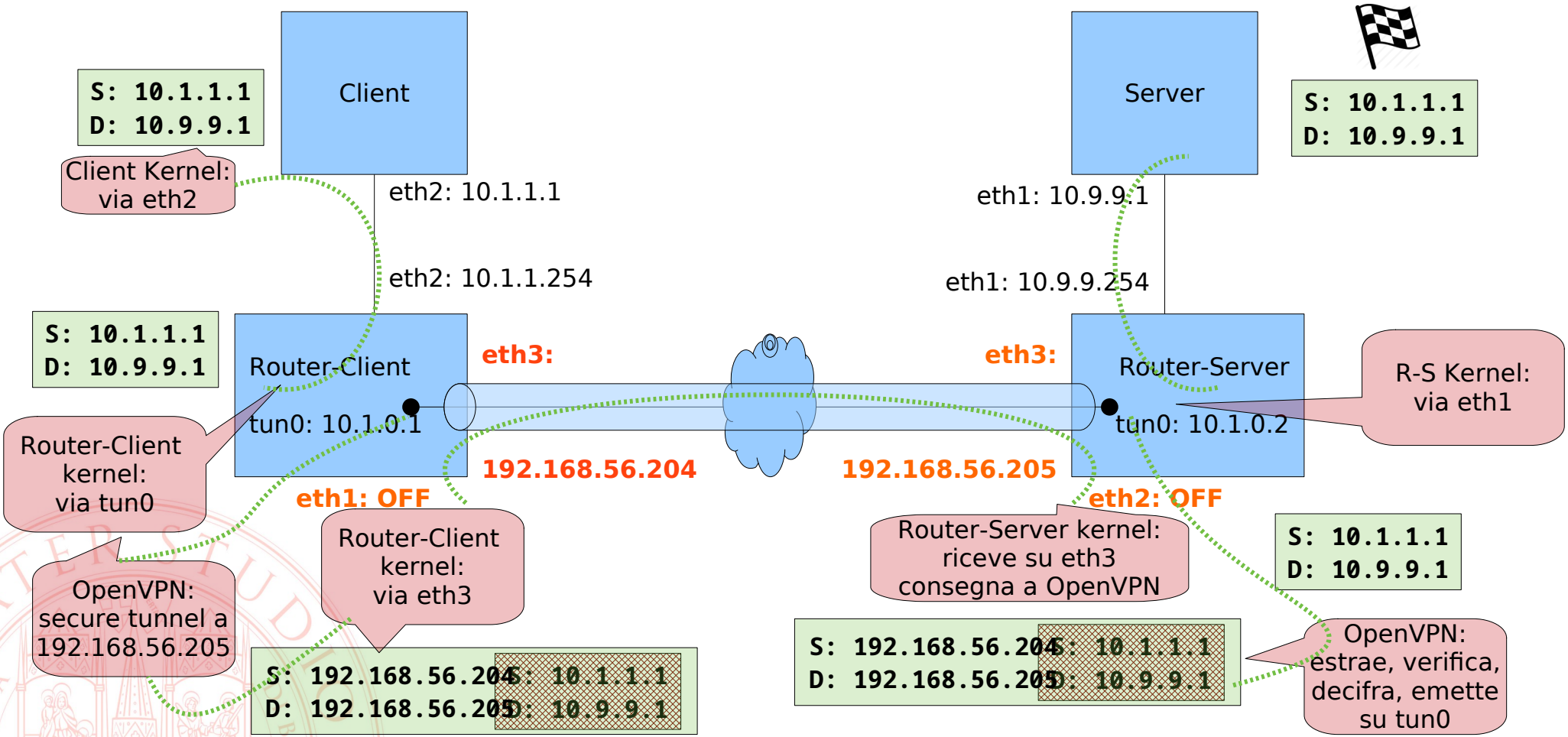
Tunnel mode

- Esempio di una rete che collega due siti remoti:



Tunnel mode

- Le due reti vedono normale instradamento IP

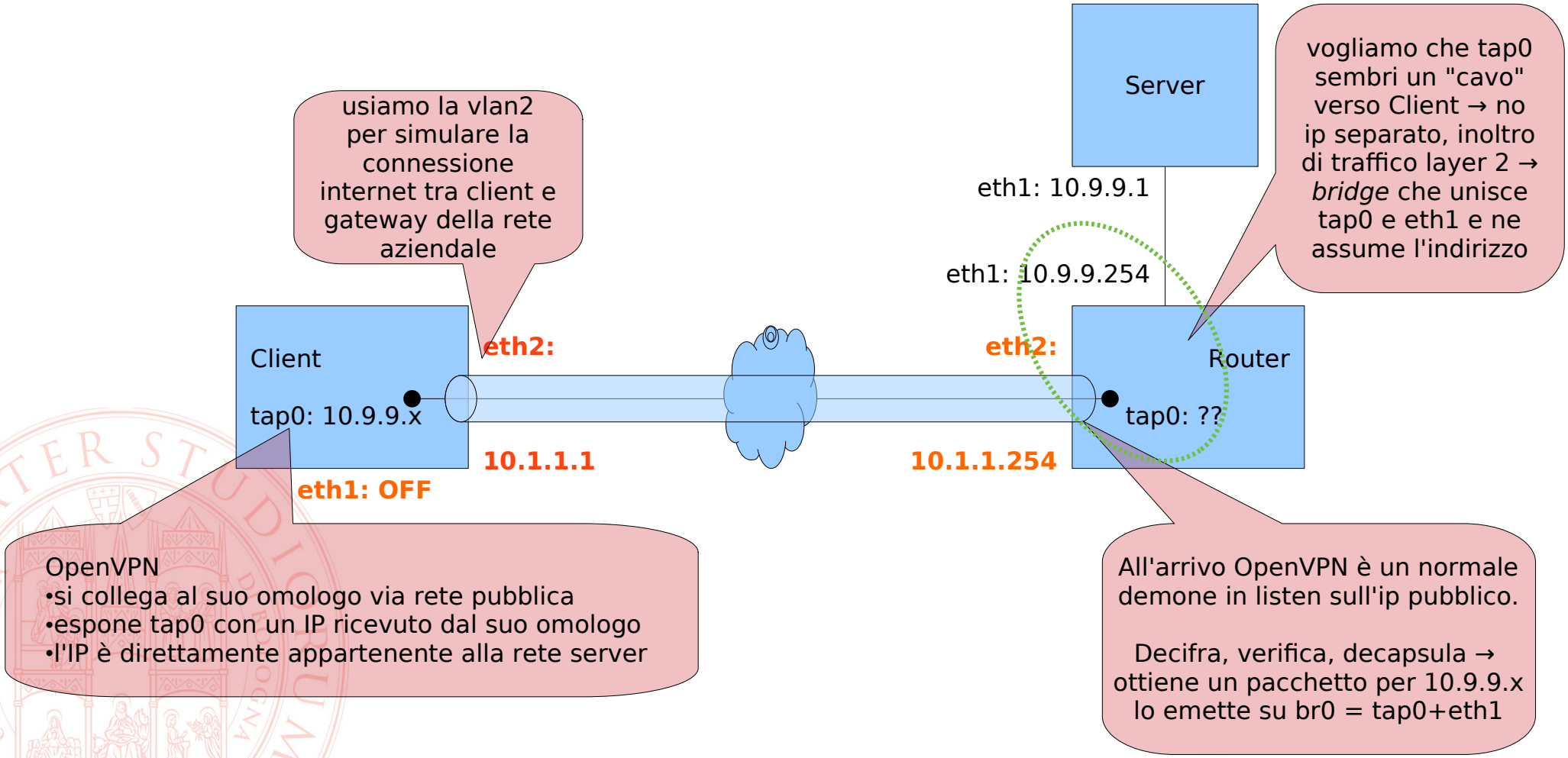


Tunnel vs. transport

- Come si vede, l'interfaccia tun è un puro artificio per creare una connessione punto-punto tra i due gateway mediata da OpenVPN
- Dal punto di vista delle applicazioni, gli indirizzi delle interfacce tun sono trasparenti e non appartengono a nessuna delle subnet effettivamente utilizzzate da client e server
- Per rendere una macchina remota virtualmente parte di una rete locale si ricorre al transport mode, tipicamente associato al *bridging*

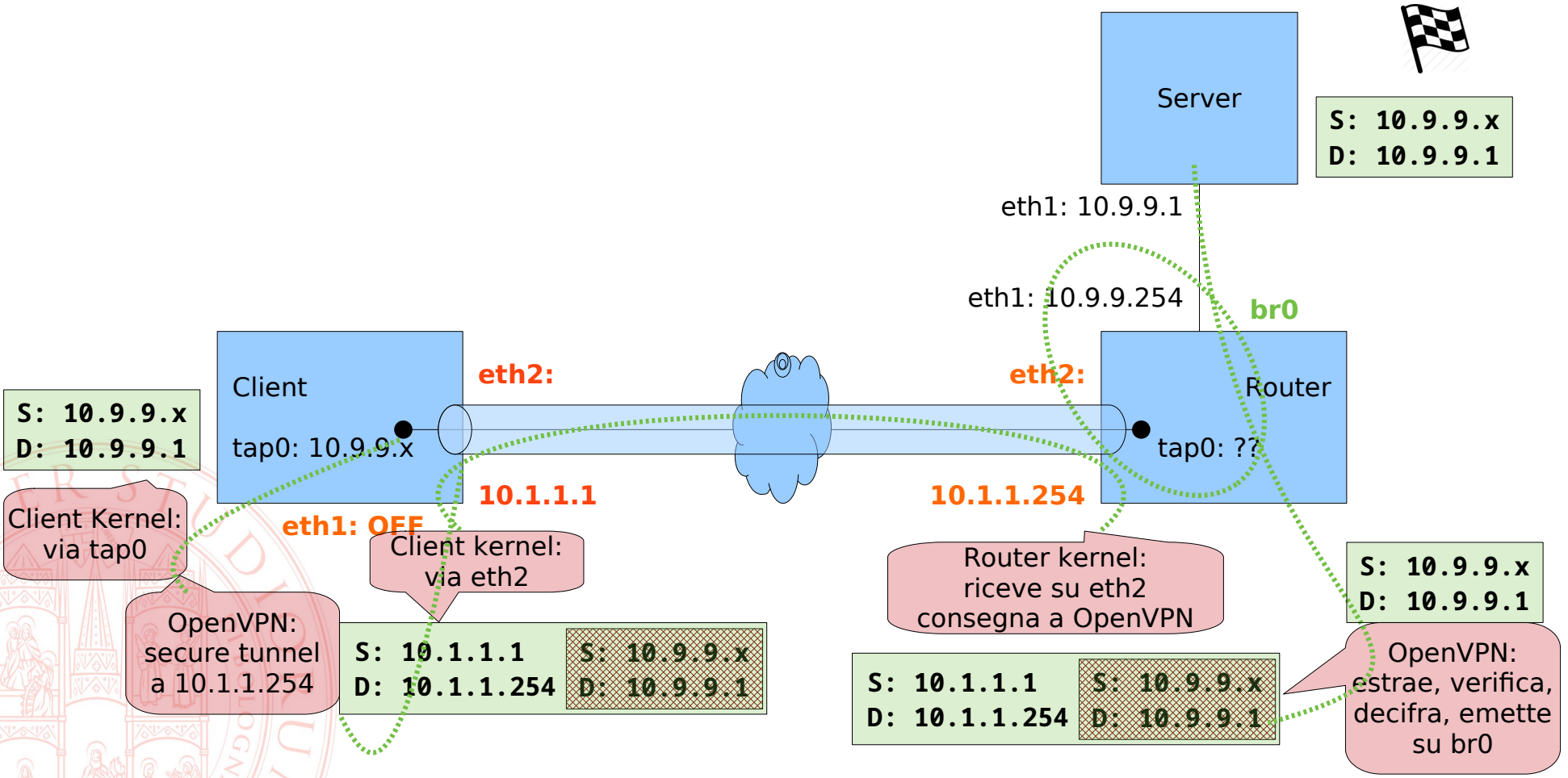
Transport mode

- Esempio di una rete che collega un host a una rete remota come se ne facesse fisicamente parte



Transport mode

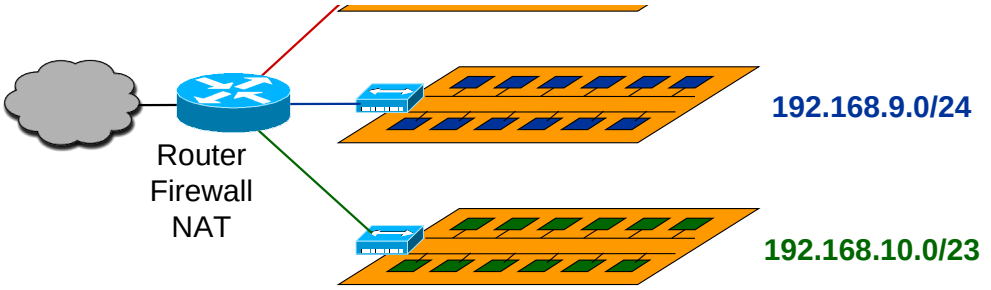
- Simuliamo una rete che collega un host a una rete remota come se ne facesse fisicamente parte



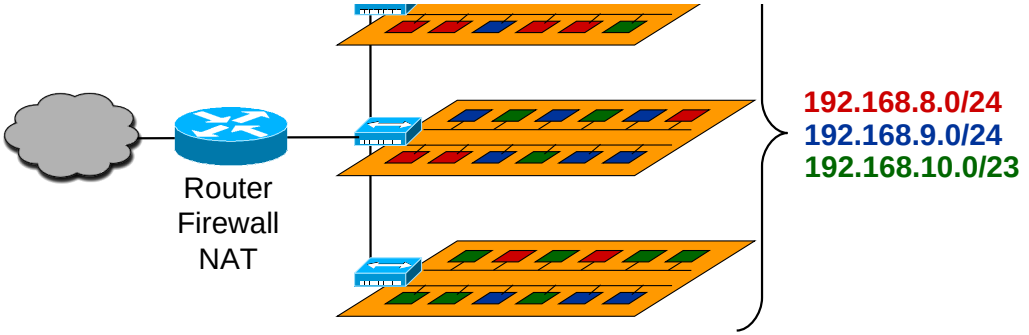
Data Link security

- Gli switch possono gestire **Virtual LAN**
  - segregano il traffico tra differenti subnet in modo che gli host di una non vedano il traffico delle altre **anche se fisicamente condividono parte dell'infrastruttura**

- Senza VLAN
  - un segmento fisico per ogni subnet
  - una porta del router per ogni subnet
  - collocazione statica delle subnet sulle LAN



- Con VLAN
  - subnet sparse su più segmenti fisici
  - anche solo una porta del router
  - collocazione configurabile dei membri delle subnet sulle LAN



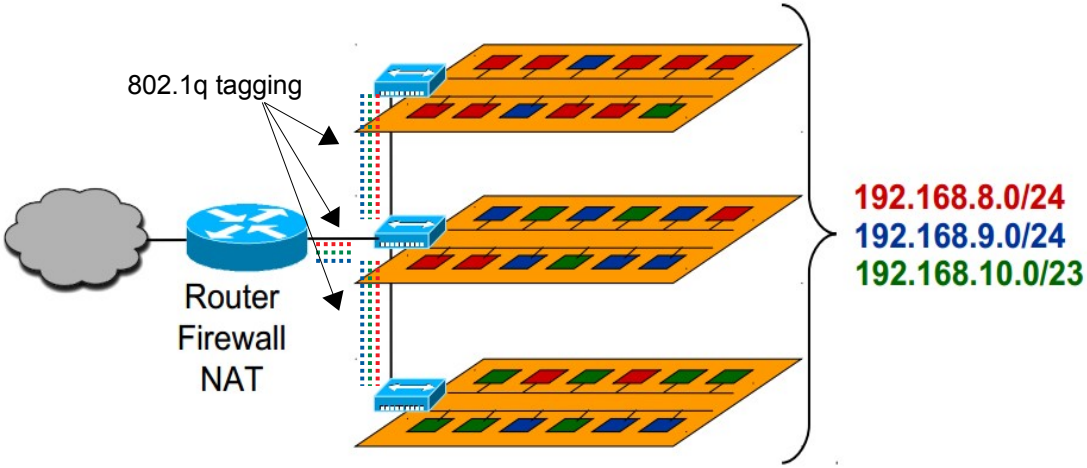
VLAN – classificazione

- VLAN statica o port-based
  - ogni porta di uno switch appartiene a una o più VLAN
  - un host può appartenere a una o più VLAN, solo in base alla porta dello switch a cui è connesso
  - per spostare un host da una VLAN a un'altra, bisogna riconfigurare la porta dello switch
- VLAN dinamica
  - un host appartiene a una o più VLAN in funzione del proprio MAC o IP, indipendentemente dalla porta dello switch a cui è connesso
  - per spostare un host da una VLAN a un'altra, bisogna riconfigurare la mappatura indirizzo-VLAN

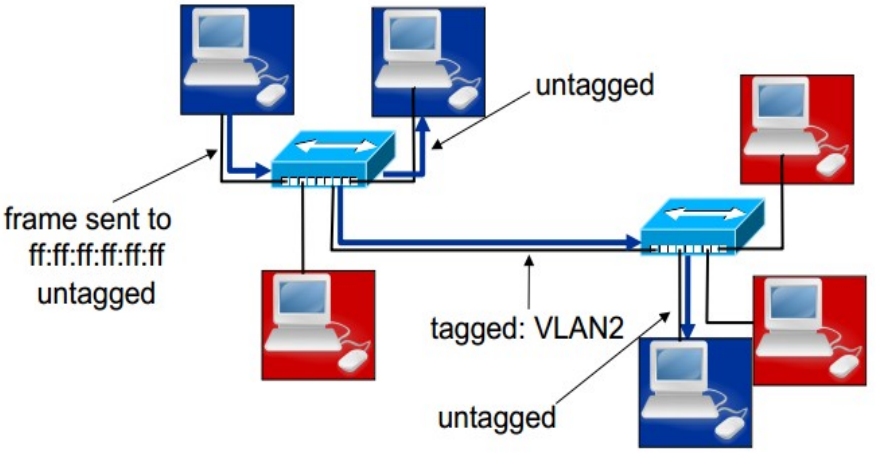
VLAN – tagging

- I pacchetti possono essere marcati con un tag
  - Frame modificato (802.1q)

- Possibilità di dare accesso a più VLAN a un dispositivo che supporti 802.1q
  - es. router



- **Possibilità di trasportare pacchetti di VLAN diverse in modo riconoscibile** tra router
  - che poi possono rimuovere il tag per destinare i pacchetti a host ignari di 802.1q

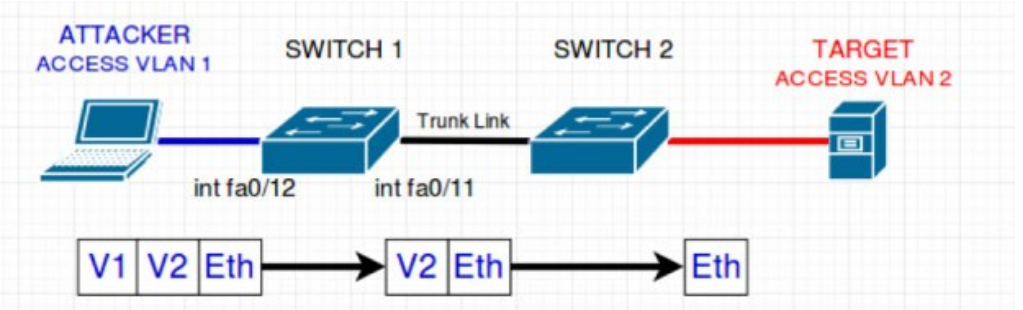


VLAN – modi di funzionamento delle porte

- Porte in access mode
  - appartengono a una singola VLAN
  - tagging dei pacchetti non necessario
  - tipico uso per connessione di semplici host
- Porte in trunk mode
  - appartengono a più VLAN
  - tagging necessario per distinguere a che VLAN appartiene ogni pacchetto
  - possono essere configurate simultaneamente per gestire pacchetti untagged, che vengono considerati appartenenti a una VLAN nativa + pacchetti tagged di altre VLAN
  - tipico uso per connessione a router o tra switch

VLAN hopping

- La separazione logica tra subnet offerta da VLAN non è sempre robusta come quella fisica; esistono attacchi che permettono di scavalcare la barriera
- **Switched spoofing**:
  - Prerequisito: vittima che accetta riconfigurazione con Dynamic Trunking Protocol – DTP)
  - l'attaccante fa credere di essere lui stesso uno switch, e configura la vittima in modo che la porta a cui è connesso venga impostata come trunk su cui far passare tutte le VLAN
- **Double Tagging**:
  - Prerequisito: catena di switch con gestione non attenta di tag annidati, e attaccante legittimamente connesso alla VLAN "V1" **nativa** del trunk che interconnette gli switch
  - l'attaccante invia a un host sulla VLAN "V2" un pacchetto costruito ad arte perché appaia come se fosse annidato: con tag V2 incapsulato in un tag V1
  - il primo switch rimuove il tag V1 e inoltra il pacchetto a tutte le porte access o native che appartengono a V1, incluso il trunk
  - il secondo switch riceve il pacchetto e lo interpreta come appartenente a V2, quindi lo inoltra all'host vittima
  - **NOTA: funziona solo in andata, non c'è modo di forzare la vittima a generare una risposta in modo da riceverla**



https://cybersecurity.att.com/blogs/security-essentials/vlan-hopping-and-mitigation

“Uso quotidiano”

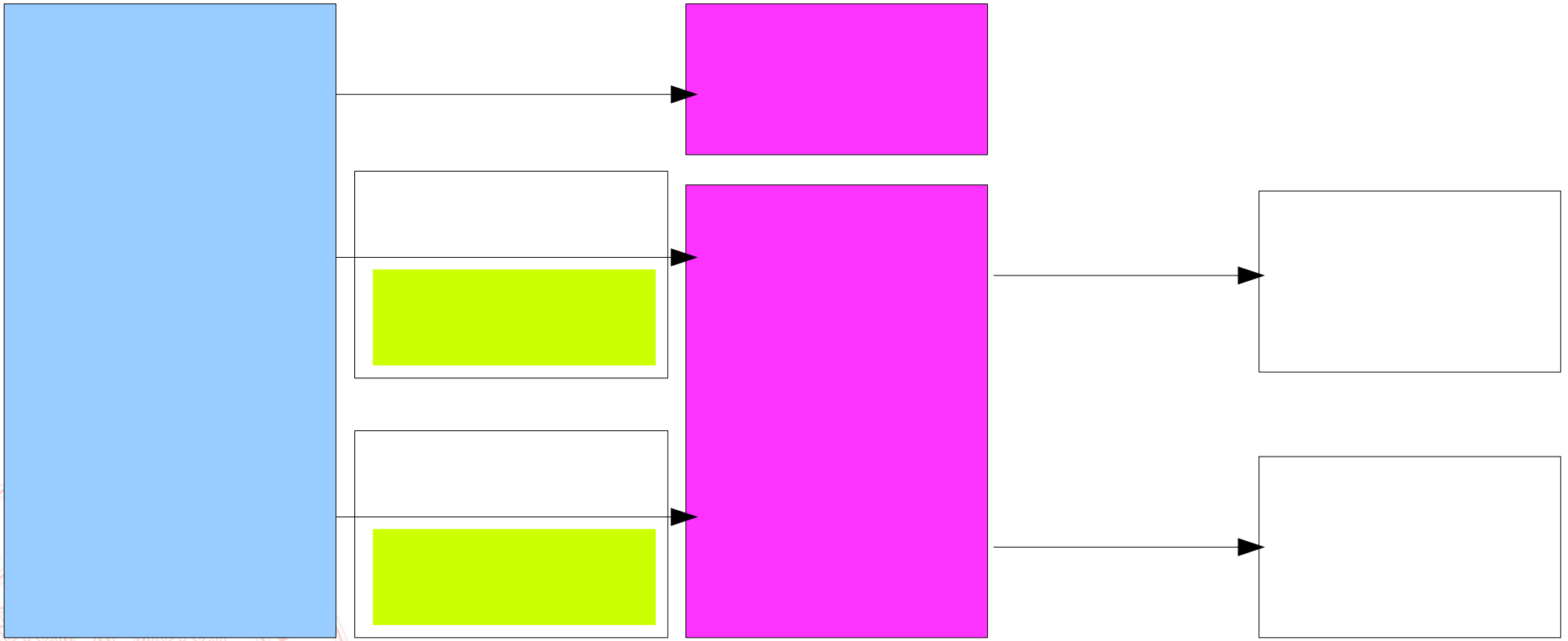
- Navigazione “anonima” con Tor
- Secure Shell
  - amministrazione remota
  - port forwarding

Preludio: SOCKS5

- SOCKS5 – RFC 1928 – può essere classificato tra i *circuit level gateway*, una tipologia di *proxy* o *firewall*
  - dispositivi che inoltrano il traffico spezzando la connessione a livello di sessione: diventano endpoint del traffico, non intermediari trasparenti come i router
- Utilizzo tipico
  - Determinare quali connessioni sono ammissibili da una rete protetta verso l'esterno
- Vantaggi
  - Può essere configurato trasparentemente agli utenti per autorizzare le connessioni da determinati host considerati fidati
  - Può agire da intermediario generico, senza bisogno di predefinire quali protocolli applicativi gestire
  - Può essere usato in combinazione con le applicazioni per differenziare le politiche sulla base degli utenti
- Svantaggi
  - Richiede la modifica dello stack dei client o la consapevole configurazione delle applicazioni

Preludio: SOCKS5

Esempio – pacchetti di protocolli di livello > 5 sono trasportati dal client al CLG dentro pacchetti SOCKS5, estratti, e inviati su di una connessione TCP/IP alla destinazione finale.

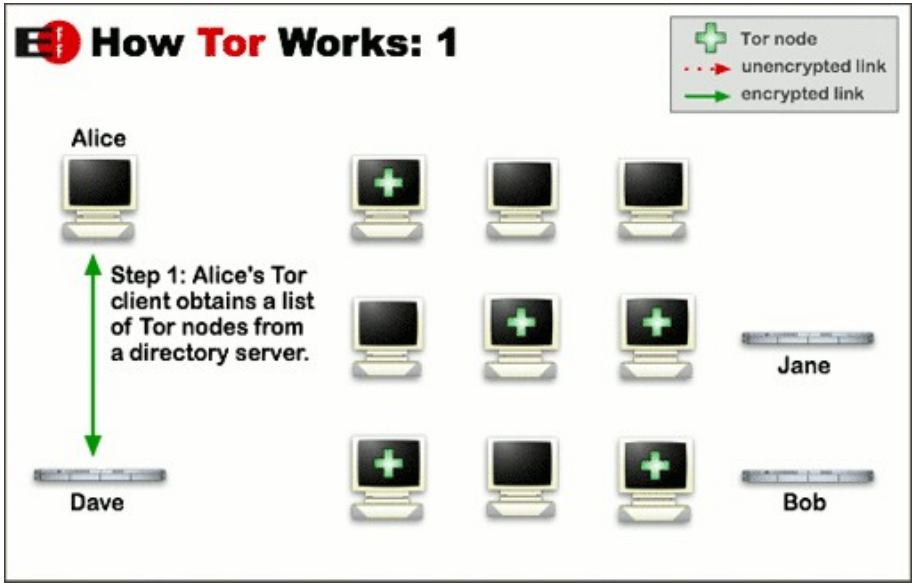


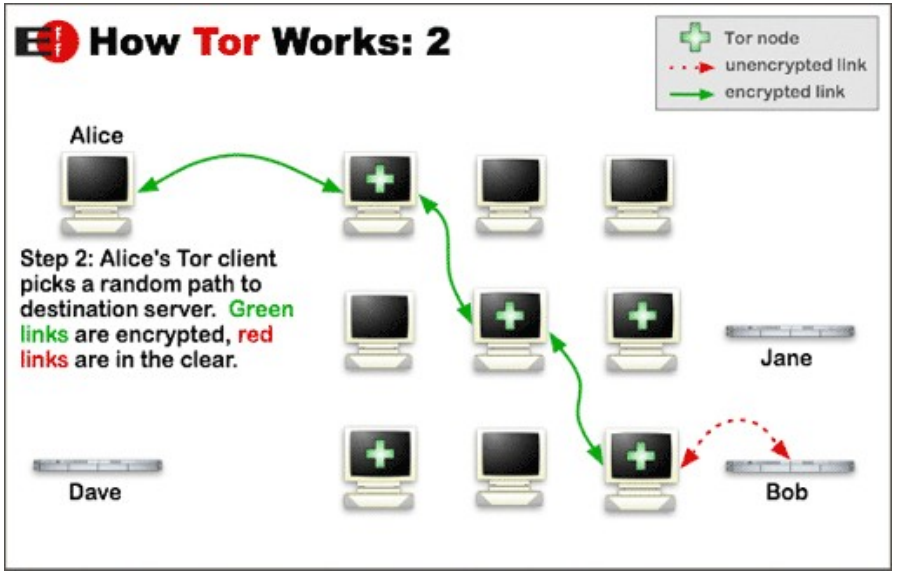
https://securityintelligence.com/posts/socks-proxy-primer-what-is-socks5-and-why-should-you-use-it/

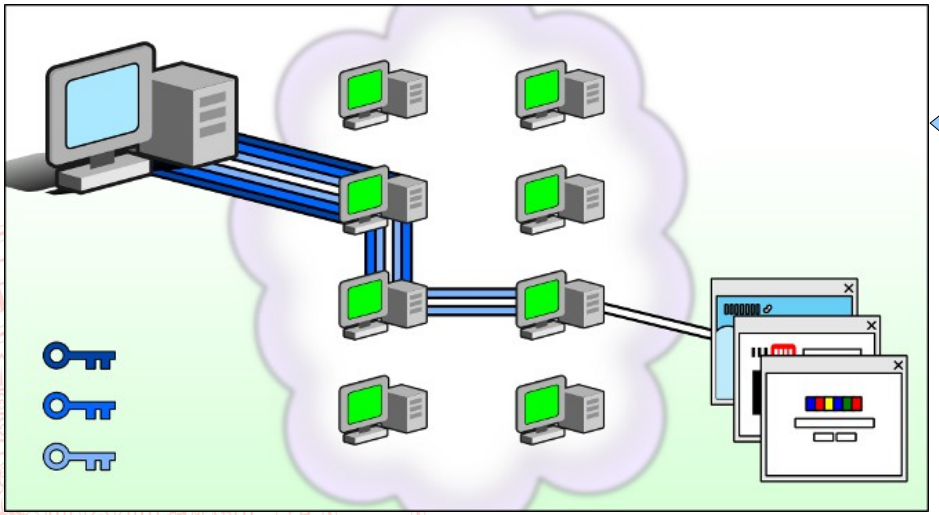
Tor

- nome originale: The Onion Router
  - Progetto open source avviato dalla Electronic Frontier Foundation (EFF)
  - Sponsorizzato tra gli altri da Google, Mozilla, SRI, NSF via diverse università USA, ...
    - e migliaia di utenti che forniscono supporto infrastrutturale
- Il protocollo di Tor permette di realizzare connessioni cifrate in cui il legame tra chi effettua richieste e il contenuto delle stesse è profondamente oscurato
- Esistono applicazioni “local proxy” che espongono un’interfaccia SOCKS5 a qualsiasi client locale per farlo accedere a TOR

Tor







- Il setup del percorso restituisce al client un set di chiavi AES condivise con ognuno dei relay attraversati
- Il messaggio è cifrato "a cipolla"
- Ogni relay conosce solo i suoi due vicini di percorso (anche in fase di costruzione)

https://www.torproject.org/about/overview

Tor

- Debolezze
  - Entry ed exit node nello stesso AS → correlazione
  - Exit node vede traffico in chiaro (ma non IP sorgente)
    - Nel payload potrebbero esserci dati ben più identificativi!
  - Bad apple → un'applicazione insicura (IP leak) porta al tracciamento anche di quelle sicure dello stesso utente
  - Uso di Tor = aumento del sospetto da parte di autorità
- Contromisure intrinseche
  - La scelta random di un percorso per ogni connessione minimizza il rischio di attraversare nodi compromessi
- Ulteriori accorgimenti
  - L'uso di cifratura applicativa oscura il contenuto anche dell'ultimo hop https://www.eff.org/it/pages/tor-and-https
  - *Bridges* = entry nodes non elencati nella directory Tor, per non mostrare all'ISP che si usa Tor (o per aggirare il suo blocco) https://bridges.torproject.org/

Secure Shell

- Necessità: amministrazione remota
- Predecessori: TELNET
  - Nessuna confidenzialità del canale
  - Nessuna autenticazione dell'host
  - Autenticazione passiva dell'utente

Secure Shell

- Il collegamento SSH tra client (ssh) e server (sshd) avviene attraverso questi passi essenziali
  - Negoziazione dei cifrari disponibili
  - Autenticazione dell'host remoto per mezzo della sua chiave pubblica
  - Inizializzazione di un canale di comunicazione cifrato
  - Negoziazione dei metodi disponibili per l'autenticazione dell'utente
  - Autenticazione dell'utente
- Ognuno dei passi elencati può essere portato a termine in modo configurabile, al fine di garantire il compromesso tra sicurezza e flessibilità più adatto al contesto.

Secure Shell – host authentication

- L'autenticazione dell'host remoto è importante per evitare di cadere nella trappola tesa da un eventuale uomo nel mezzo, che potrebbe così catturare la password dell'amministratore spacciandosi per l'host su cui egli vuole effettuare il login
  - Non è previsto un sistema centralizzato di attestazione dell'autenticità della chiave dell'host
    - solo supporto non ufficiale a X.509
  - Alla prima connessione l'amministratore deve utilizzare un metodo out-of-band per determinare la correttezza della chiave pubblica presentata dall'host
  - Alle connessioni successive la chiave pubblica memorizzata dal client dell'amministratore permette di effettuare un'autenticazione attiva

- Le chiavi pubbliche vengono memorizzate nel file known_hosts nella directory .ssh posta nella home dell'utente sul client.

Secure Shell – user authentication

- Ci sono due possibilità per l'autenticazione dell'utente sull'host remoto
  - Autenticazione passiva, tradizionale, con username e password – i dati sono trasmessi all'host autenticato su di un canale cifrato, quindi con buon livello di sicurezza
  - Autenticazione attiva, per mezzo di un protocollo challenge-response a chiave pubblica – presuppone che l'utente si doti della coppia di chiavi, e che installi correttamente sull'host remoto la chiave pubblica

Secure Shell – user authentication

- In entrambi i casi, l'identità dell'utente con cui viene tentato il login sull'host remoto può essere selezionata
  - in assenza di indicazioni specifiche verrà usato lo stesso nome utente con cui l'operatore sta lavorando sul client

  Es:
  - utente `marco` sul client esegue `ssh remoteserver`

    → si presenta come utente `marco` su `remoteserver` e si deve autenticare di conseguenza
  - utente `marco` sul client esegue `ssh root@remoteserver`

    → si presenta come utente `root` su `remoteserver` e si deve autenticare di conseguenza

Secure Shell – key generation

- Per poter effettuare l'autenticazione attiva un utente deve
  - generare una coppia di chiavi asimmetriche
    - es. `ssh-keygen -t rsa -b 2048`
      - chiave privata `.ssh/id_rsa`
      - chiave pubblica `.ssh/id_rsa.pub`
  - installare sull'host remoto la chiave pubblica.
    - tool di copia

      `ssh-copy-id [-i file] user@remote`
    - oppure
      - copia manuale su host remoto

      `scp .ssh/id_rsa.pub user@remote:`
      - append alla lista di utenti autorizzati (su *remote*)

      `cat id_rsa.pub >> .ssh/authorized_keys`

Secure Shell – avvertenze

Il ruolo autenticante della password viene sostituito dalla presenza della chiave privata dell'utente sul client – la segretezza della password è quindi sostituita dalla riservatezza del file che contiene la chiave privata

- Grande cura nell'impostazione dei permessi di file e directory (nota di tipo pratico: spesso il passwordless login non funziona semplicemente perché i permessi sulla directory *.ssh* dell'host remoto sono troppo larghi, e quindi il server sshd "non si fida" dell'integrità del suo contenuto)
- Possibilità di proteggere la chiave privata con una password

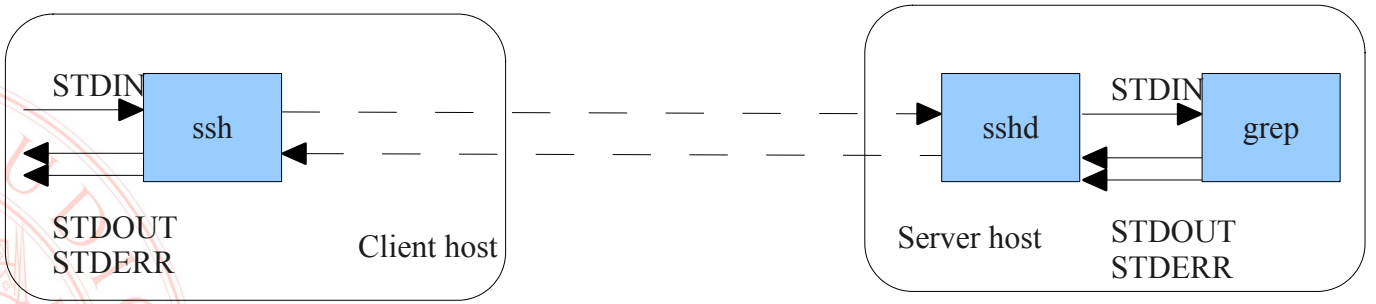
– Vi priva della possibilità di passwordless login
– Più sicuro comunque che utilizzare direttamente la password dell'account remoto, e più pratico se si amministrano molti host remoti

Secure Shell – esecuzione remota

- Lanciando `ssh utente@host` si ottiene un *terminale remoto* interattivo.
- Aggiungendo un ulteriore parametro, viene interpretato come **comando** da eseguire sull'host remoto al posto della shell interattiva; gli stream di I/O di tale comando vengono riportati attraverso il canale cifrato sul client.

  Es: `ssh root@server "grep pattern"`

  - I dati forniti attraverso STDIN al processo ssh sul client vengono resi disponibili sullo STDIN del processo grep sul server
  - STDOUT e STDERR prodotti dal processo grep sul server "fuoriescono" dagli analoghi stream dal processo ssh sul client



SSH tunnelling “L”

Dalla man page:

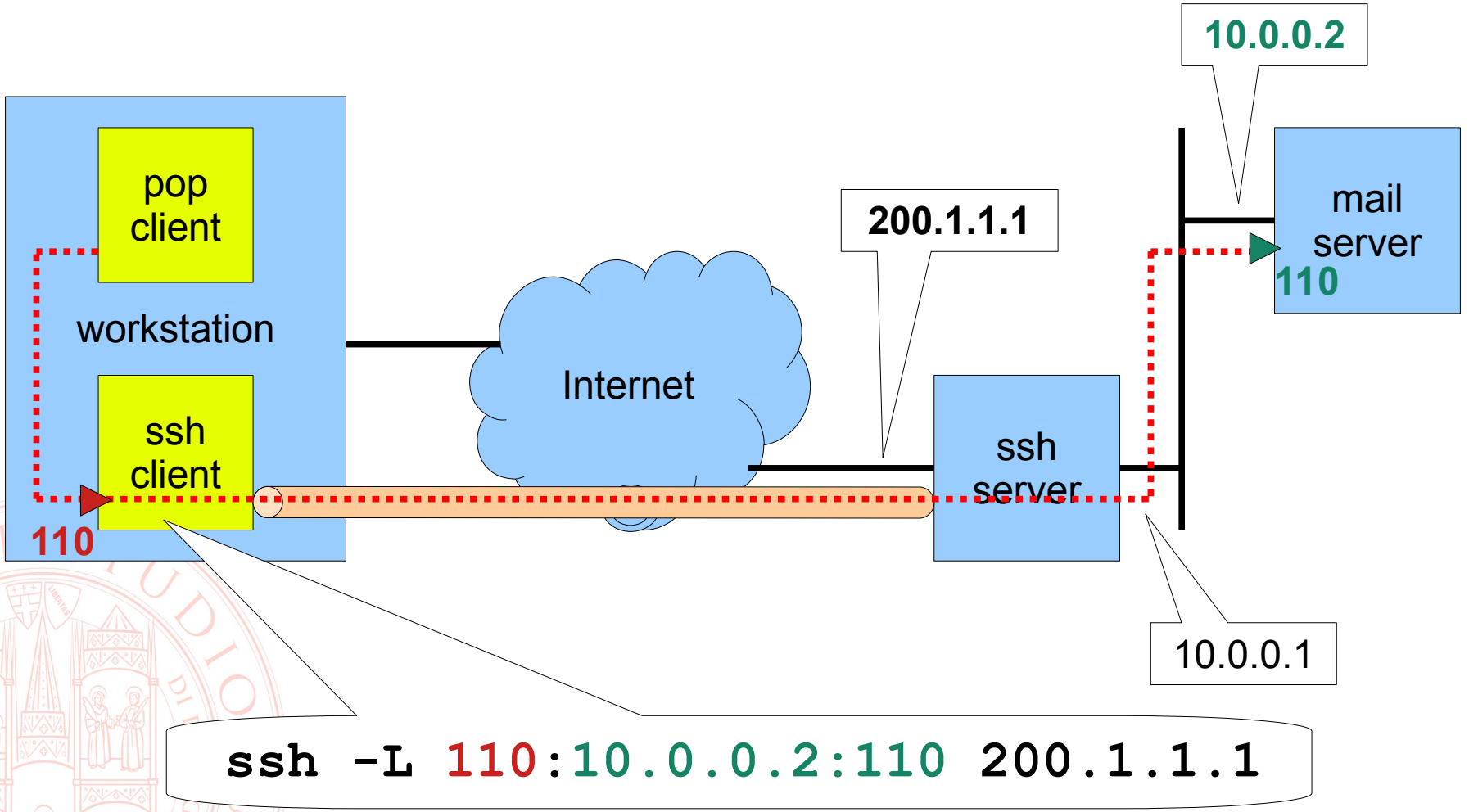
- **"poor man's VPN"**

`-L [bind_address:]port:host:hostport`

Specifica che la porta specificata sull'host locale (client) deve essere inoltrata all'host e alla porta host specificati sul lato remoto. Funziona allocando un socket per ascoltare la porta sul lato locale, facoltativamente associato al bind_address specificato. Ogni volta che viene effettuata una connessione a questa porta, la connessione viene inoltrata sul canale protetto e viene stabilita una connessione alla porta host hostport dalla macchina remota.

- **Prerequisiti: dare a un utente accesso SSH a un gateway “di frontiera” per l’organizzazione**
  - **Autenticazione forte**
  - **Possibilità di restringere le operazioni ammissibili**
- **Effetto: rendere raggiungibili host e servizi al di là del gateway**

SSH tunnelling “L” – esempio



SSH tunnelling “R”

Dalla man page:

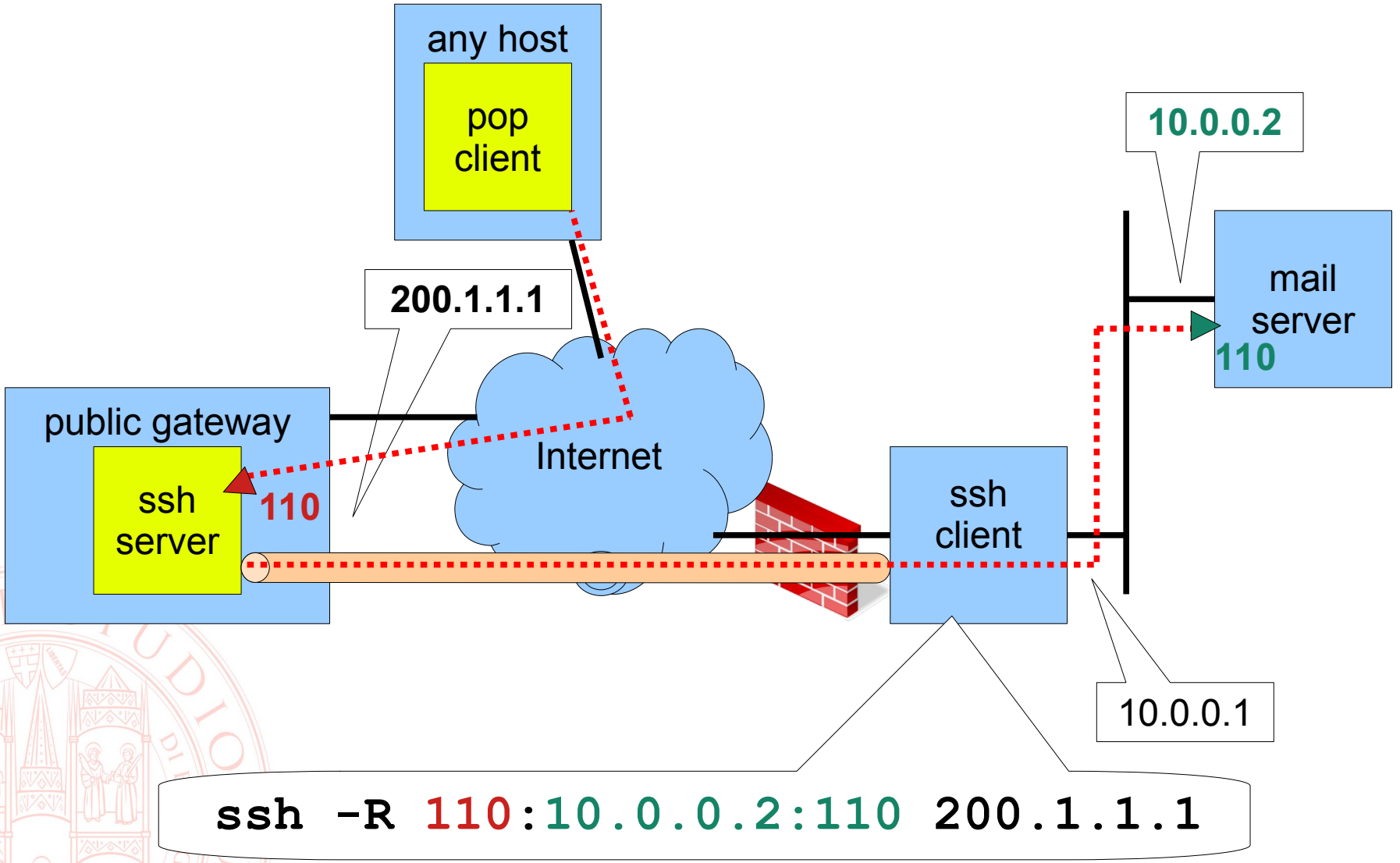
- **creare un accesso pubblico a una rete privata**

`-R port:host:hostport`

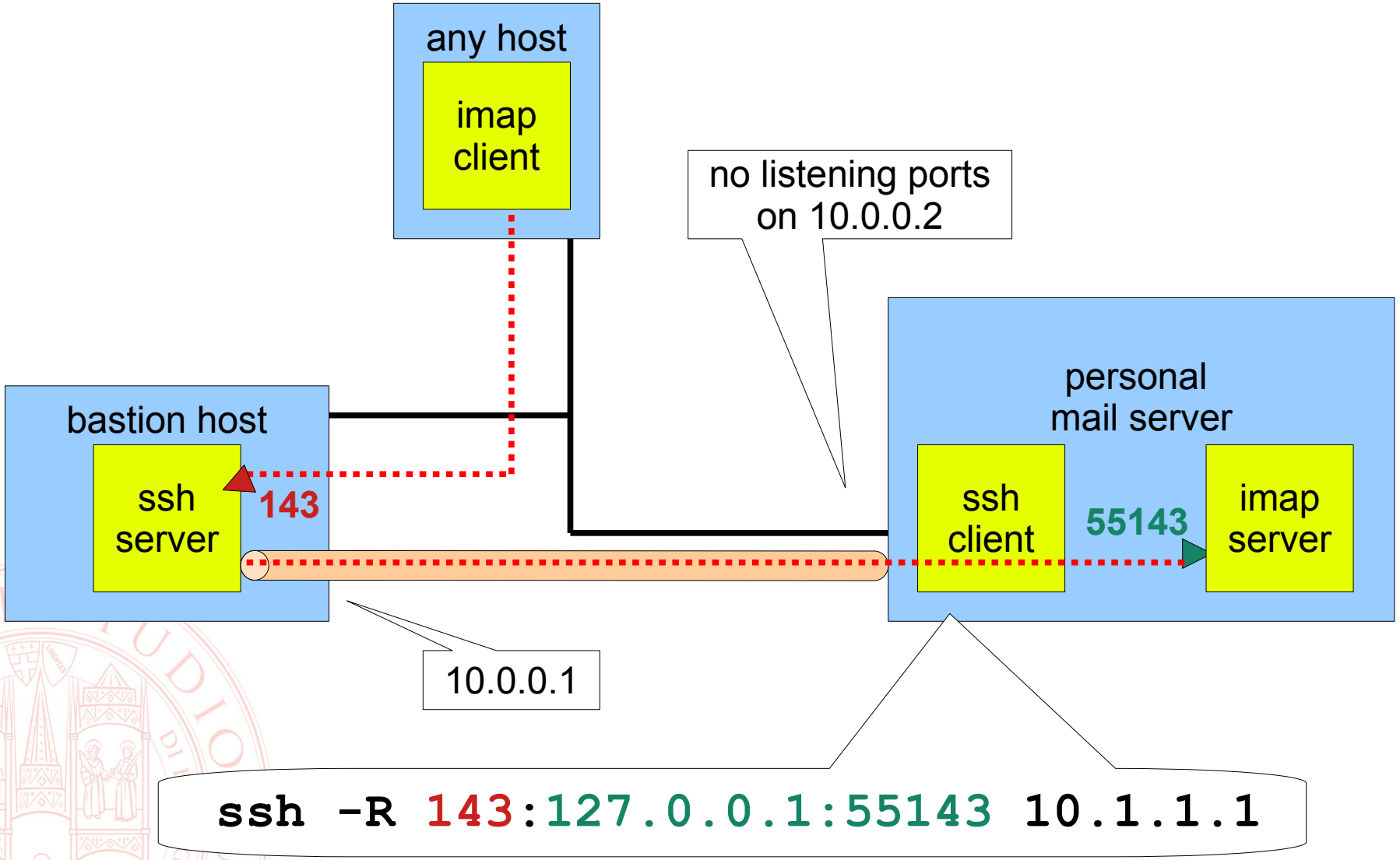
Specifica che la porta specificata sull'host remoto (server) deve essere inoltrata all'host e alla porta host specificati sul lato locale. Funziona allocando un socket per ascoltare la porta sul lato remoto e ogni volta che viene effettuata una connessione a questa porta, la connessione viene inoltrata sul canale protetto e viene stabilita una connessione alla porta host hostport dalla macchina locale.

- **Prerequisiti: avere a disposizione un gateway (con almeno un lato) esterno alla rete privata (pubblicamente) raggiungibile**
- **Effetto: rendere raggiungibile dalla rete esterna (non necessariamente pubblica)**
  - **un servizio locale non esposto sulla rete privata**
  - **un servizio della rete privata (inaccessibile dall’esterno)**

SSH tunnelling “R” – esempio “buca firewall”



SSH tunnelling “R” – esempio “filtro locale”



SSH tunnelling “D”

Dalla man page:

- **attivazione di un proxy SOCKS**

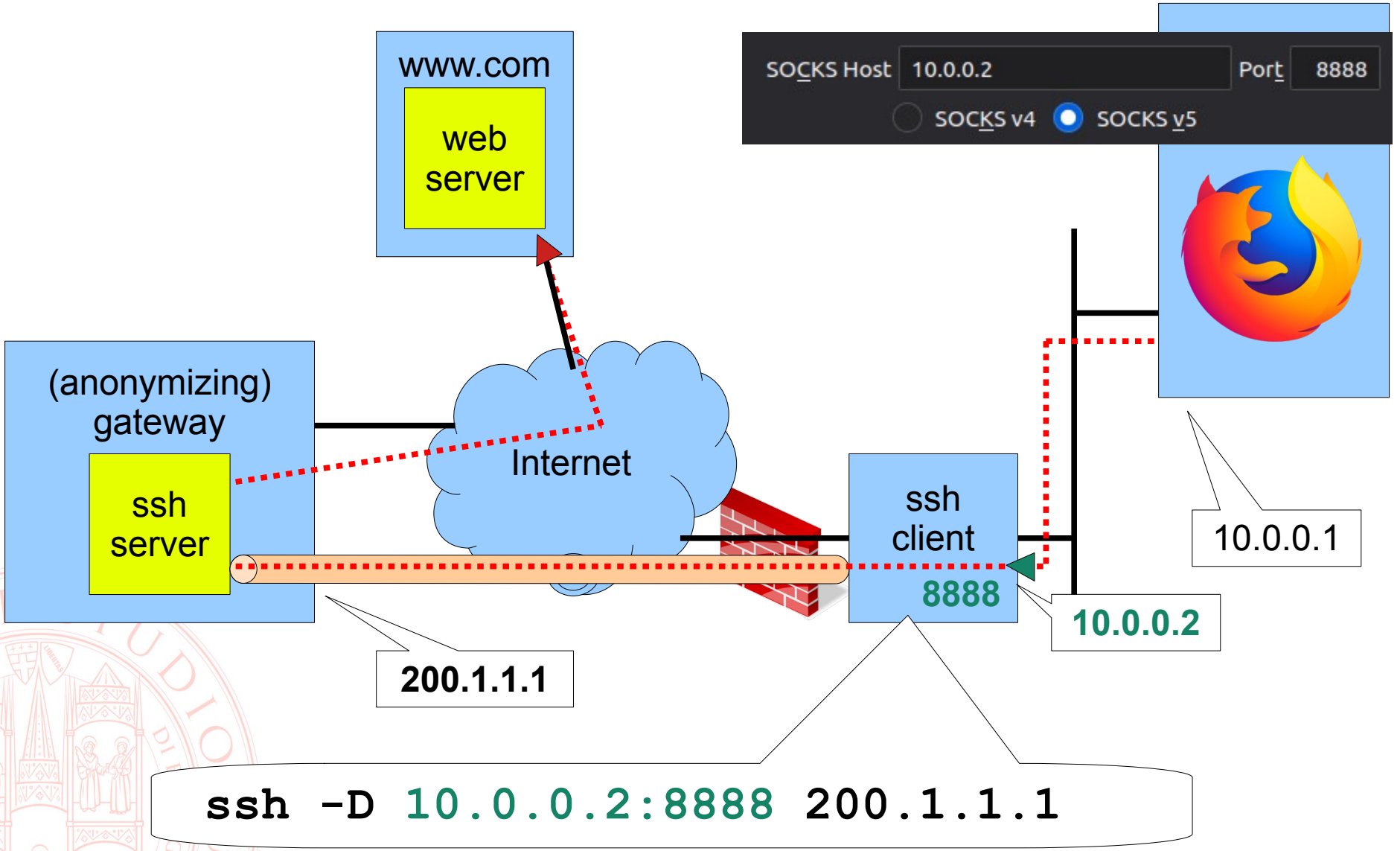
`-D [bind_address:]port`

Specifica un port forwarding "dinamico" a livello di applicazione locale.

Funziona allocando un socket per ascoltare la porta sul lato client, facoltativamente associata al bind_address specificato. Ogni volta che viene stabilita una connessione a questa porta, la connessione viene inoltrata sul canale protetto e là viene utilizzato il protocollo dell'applicazione per determinare dove connettersi dalla macchina remota. Attualmente i protocolli SOCKS4 e SOCKS5 sono supportati e ssh agirà come server SOCKS.

- **Prerequisiti: avere a disposizione un gateway “sensato” da cui far apparentemente originare**
- **Effetto: simil-ToR (senza l’elusione del triplo salto), o simil-”L” ma rendendo accessibili porte arbitrarie, un po’ come una VPN**

SSH tunnelling “D”



SSH tunnelling “J”

Dalla man page:

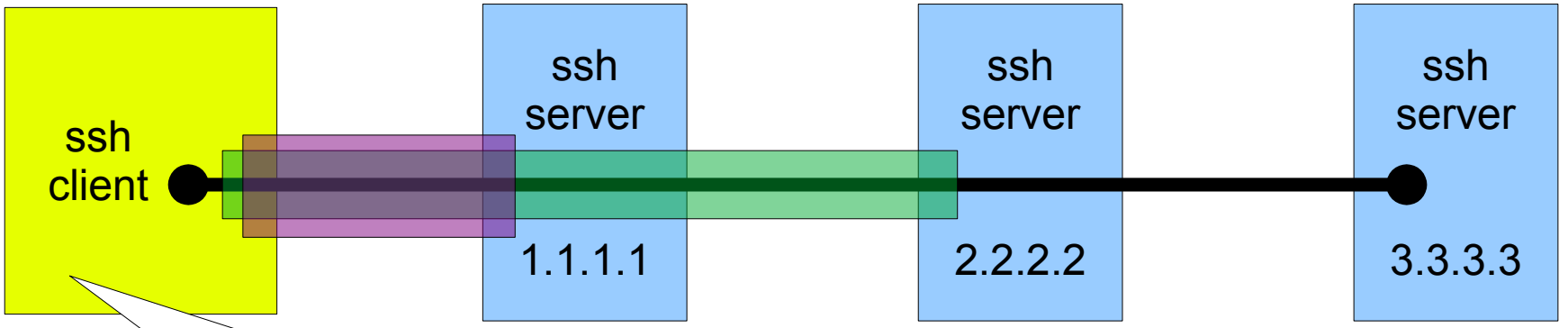
- **creare un accesso pubblico a una rete privata**

`-J jumphost`

Connette via SSH all'host di destinazione effettuando prima una connessione SSH al jumphost passato come parametro. È possibile specificare molteplici salti separati da virgole.

- **Prerequisiti: avere a disposizione un gateway raggiungibile dal client e dal quale si possa raggiungere la destinazione finale**
- **Effetto: come “L” ma specifico per SSH, e multi-salto**

SSH tunnelling “J” – esempio




```
ssh  -J  u1@1.1.1.1,u2@2.2.2.2  u3@3.3.3.3
```